Coarse-grained modelling for soft matter scattering

submitted by

Andrew R. McCluskey

for the degree of Doctor of Philosophy

of the

UNIVERSITY OF BATH

Department of Chemistry

March, 2019

Declaration of Authorship

I, Andrew R. McCluskey, declare that this thesis titled, "Coarse-grained modelling for soft matter scattering" and the work presented in it are my own. I confirm that:

- where the thesis or any part of the thesis such as a published paper, has been produced jointly with others, that a substantial part is the original work of myself, and
- where the thesis incorporates material already submitted for another degree, the extent of that material and the degree, if any, obtained.

Signed:

Date:

"Atticus told me to delete the adjectives and I'd have the facts."

Scout Finch – To Kill a Mockingbird

UNIVERSITY OF BATH

Abstract

Department of Chemistry

Doctor of Philosophy

Coarse-grained modelling for soft matter scattering

by Andrew R. McCluskey

The abstract will go here. This will be a brief description of the work in the thesis.

Reproducibility Statement

This thesis exists as a piece of completely reproducible research. I have endevoured to include as much algorithmic and methodological detail within the text. However, in order to provide complete, and easy, reproducibility an electronic supplementary information (ESI) is available online in the form of a Git repository. This ESI provides full details of the analyses performed in this work and access to an automated analysis workflow.

The ESI may be accessed at the following DOI: 10.5281/zenodo.xxxxxxx.

Publications

Some of the work covered in Chapter 2 has been previously published in:

- Andrew R. McCluskey and Karen J. Edler, *Curr. Org. Chem.*, 2018, **22** (8), 750, DOI: 10.2174/1875692115666170612104439.

Some of the work covered in Chapter 3 has been previously published in:

- Andrew R. McCluskey, Adrian Sanchez-Fernandez, Karen J. Edler, Stephen C. Parker, Andrew J. Jackson, Richard A. Campbell, and Thomas Arnold, *Phys. Chem. Chem. Phys.*, 2019, **21** (11), 6133-6141, DOI: 10.1039/C9CP00203K.

Some of the work covered in Chapter 4 has been previously published in:

- Andrew R. McCluskey, James Grant, Andrew J. Smith, Jonathan L. Rawle, David J. Barlow, M. Jayne Lawrence, Stephen C. Parker, and Karen J. Edler, *J. Phys. Comm.*, 2019, Accepted, DOI: 10.1088/2399-6528/ab12a9.

Some of the work covered in Chapter 6 has been previously published in:

- Andrew R. McCluskey, Benjamin J. Morgan, Karen J. Edler, and Stephen C. Parker, *J. Open Source Educ.*, 2018, **1** (2), 19, DOI: 10.21105/jose.00019.
- Andrew R. McCluskey, James Grant, Adam R. Symington, Tim Snow, James Doutch, Benjamin J. Morgan, Stephen C. Parker, and Karen J. Edler, *J. Appl. Crystallogr.*, 2019, Submitted, arXiv: 1902.01324.

Acknowledgements

This is where I will acknowledge people. Need to remember everyone...

Contents

List of Abbreviations

BM	bending magnet
DLS	Diamond Light Source
ESRF	European Synchrotron Radiation Facility
ESS	European Spallation Source
GiSAS	grazing incidence small angle scattering
ID	insertion device
ILL	Institut Laue-Langevin
SAS	small angle scattering
ToF	time-of-flight

Physical Constants

	$\pi = 3.1415\ldots$
Planck constant	$h = 6.626\ \ldots\ \times 10^{-34}\,\mathrm{J\,s}$

List of Symbols

a_0	optimum head-group area	m^2
b	scattering length	m
m	mass	kg
n_i	refractive index	
n	number of individual q-vectors	
q	scattering vector magnitude	m^{-1}
$r_{n,n+1}$	Fresnel equation coefficient	
$res(q)$	resolution function	
t_F	time of flight	s
v	velocity	$\mathrm{m\,s}^{-1}$
B	resultant matrix	
E_k	kinetic energy	J
I	intensity	
L_F	distance of flight	m
M	layer matrix	
N	number of particles	
N_P	number of magnets	
R	reflected intensity	
S	nuclear spin	
V	volume	m^3
$\mathbf{k}_i$	incident wavevector	m^{-1}
$\mathbf{k}_f$	final wavevector	m^{-1}
$\mathbf{q}$	scattering vector	m^{-1}
$\mathbf{r}_i$	particle position	
β_c	fraction of the speed of light	
β_n	phase factor	
θ	scattering angle	rad
θ_c	critical angle	rad
θ_e	angle between electron and photon	rad
λ	wavelength	m
ρ	scattering lenght density	m^{-2}
σ	interfacial roughness	m
σ_{coh}	coherent cross-section	m^2
σ_{incoh}	incoherent cross-section	m^2
ϕ	scattering angle	rad
ω	frequency	s^{-1}
ω_i	initial frequency	s^{-1}
ω_f	final frequency	s^{-1}
Λ	temperature factor	
Φ	Golden ratio	
$\mathrm{d}\sigma(q)/\mathrm{d}\Omega$	differential cross-section	

1 Introduction

2 Theory

2.1 Probing radiation

This work is focussed on the use of X-ray and neutron scattering, therefore it is pertinent to discuss how each of these probing radiation is produced and detail the advantages of each with respect to the other.

2.1.1 Generation of X-rays

X-rays are a form of electromagnetic radiation similar to visible light, albeit with a much shorter wavelength, from $0.01\,\mathrm{nm}$ to $10\,\mathrm{nm}$. There are four common ways to produce X-rays; three are available within the laboratory, while the other is exclusive to large scale facilities.

The three laboratory source X-ray generation techniques are the X-ray tube, the rotating anode, and the liquid jet. An X-ray tube consists of a filament and an anode within a vacuum chamber, by passing a high voltage electrical current across the filament electrons are emitted which accelerate towards the anode. On collision with the anode, the rapid deceleration results in the emission of X-rays of a characteristic wavelength based on the anode material [1]. The most common material for an X-ray tube anode is copper which gives off radiation of about $8\,\mathrm{keV}$.

Another common laboratory method for the generation of X-rays is the rotating anode, which is an improvement on the X-ray tube. In the X-ray tube, each time that an electron contacts the anode there is some energy transfer, this means that over many millions of collisions, the temperature of the anode can rise significantly, leading to a temperature limitation on the X-ray flux available. This lead to the development of the rotating anode, this is simply where the anode is made from a rotating wheel, so that the bombardment is spread across the whole wheel reducing the energy localisation. This allows an increase in the photon flux by about an order of magnitude [1].

The final laboratory method for X-ray generation is the liquid jet source (branded MetalJet by excillum) [2]. For the liquid jet X-ray source, an electron beam is incident on a liquid metal sample (usually gallium or indium alloy), rather than traditional solid metal. This means that the electron intensity and therefore X-ray brightness, available to the liquid jet source is much greater than a rotating anode source.

The final method of X-ray generation is at a synchrotron facility, this method has the drawback that it requires access to a national or international facility; such as Diamond Light Source (DLS) or the European Synchrotron Radiation Facility (ESRF). The way in which X-rays are generated at the synchrotron involves the acceleration of an electron, rather than the deceleration as with the laboratory sources. This is achieved by having relativistic electrons travel around a curve, from Newtonian mechanics it is known that travelling on a curve at constant speed is equivalent to acceleration. First electrons are accelerated, after being

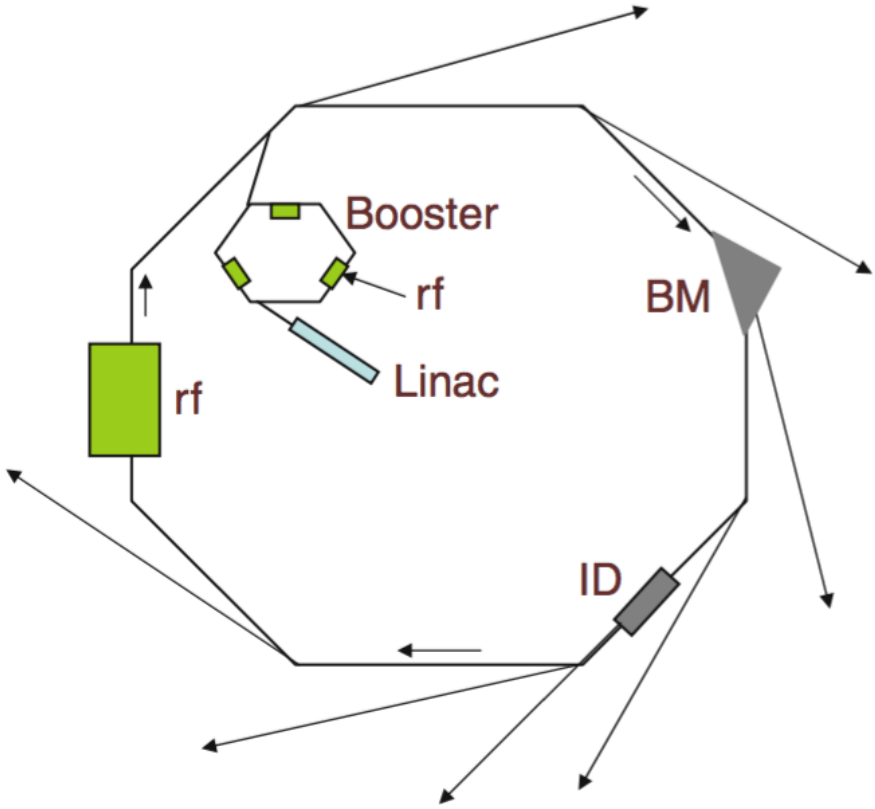


FIGURE 2.1: A schematic representation of a synchrotron radiation source, identifying the Linac, the booster ring, the radio-frequency cavities (rf), the bending magnet (BM) and the insertion device (ID), from Reference [3]. Reprinted/adapted by permission from Springer Nature Customer Service Centre GmbH: Springer Nature Bases of Synchrotron Radiation, Light Sources, and Features of X- Ray Scattering Beamlines by M.C. García-Gutiérrez, D.R. Rueda© (2009).

produced in a linear accelerator (Linac), to near the speed of light in a booster synchrotron before injecting them into the storage ring. In the storage ring, the electrons are kept at relativistic speeds with bending magnets (BM) and straight sections making up a ring (Figure 2.1). The circularity of the ring is dependent on the number of bending magnets that make up the ring; for example, DLS had 48 bending magnets with 48 straight sections at the time of construction.

When an electron accelerates (or travels on a curve), Cherenkov radiation is emitted in accordance with the Cherenkov relation,

$$n_i \beta_c \cos\theta_e = 1, \tag{2.1}$$

where, n_i is the refractive index for the dielectric medium, β_c is the fraction of the speed of light at which that electron is travelling, and θ_e is the angle between the electron trajectory and the trajectory of the resulting photon [3]. The curve is the result of a bending magnet, meaning that at each bending magnet there can be a beamline which gives out synchrotron light. The light that is given off from a bending magnet is continuous and broad, covering a wide range of the electromagnetic spectrum. The alternative to a bending magnet beamline is a beamline which is served by an insertion device (ID). An insertion device is able to offer

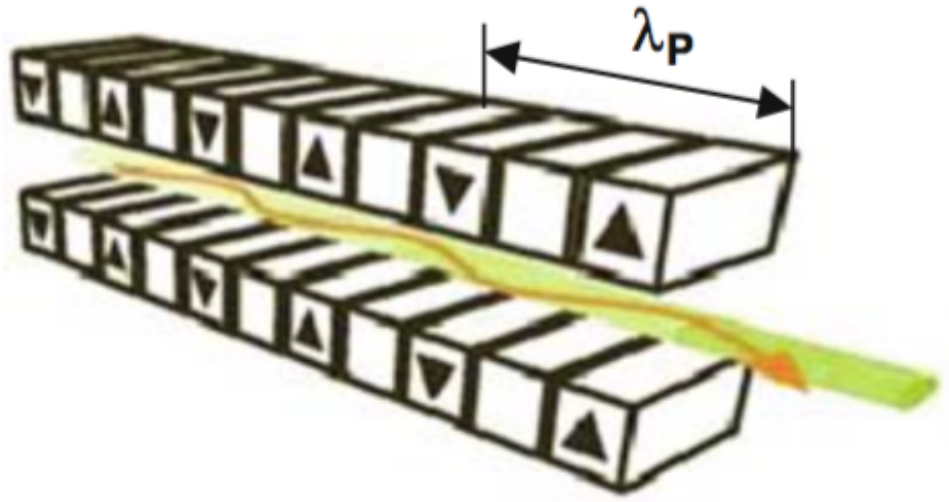


FIGURE 2.2: A diagram of an undulator insertion device such as that on I07 or I22 where λ_P is the period length between opposing magnets, from Reference [3]. Reprinted/adapted by permission from Springer Nature Customer Service Centre GmbH: Springer Nature Bases of Synchrotron Radiation, Light Sources, and Features of X- Ray Scattering Beamlines by M.C. García-Gutiérrez, D.R. Rueda© (2009).

TABLE 2.1: A comparision of the photon brilliance from different light sources. Adapted, with permission of Oxford University Press©, from Reference [4].

Light source	Approximate brilliance/ $\mathrm{photons\,s^{-1}mrad^{-2}0.1\%bandwidth^{-1}}$
Candle	10^5
X-ray tube	10^8
Sun	10^{10}
Bending magnet	10^{15}
Undulator	10^{20}

more specific radiation characteristics (photon energy, narrower band) than a bending magnet, and are placed on the straight sections of the synchrotron. Common insertion devices include wavelength shifters, wigglers, and undulators.

The type of insertion device that is present at both I07 and I22 at DLS is an undulator. An undulator consists of a series of magnets of opposing polarity which causes the electrons to 'wiggle' back and forth (Figure 2.2). This results in a superposition of radiation from N_P sources, where N_P is the number of magnets, yielding quasi-monochromatic radiation. The brilliance of different X-ray sources are compared in Table 2.1, this shows the significant benefit that an undulator offers in terms of photon brilliance.

2.1.2 Generation of neutrons

Neutrons hold an advantage over X-rays, particularly for application to the study of soft matter, in the ability to utilise contrast variation to increase the quantity of information from

the sample, this is discussed in detail in Section 2.2.9. However, neutrons cannot be produced safely on a laboratory scale, therefore it is always necessary to visit large scale facilities to harness neutrons for scattering experiments. These facilities come in two flavours; the reactor source and the spallation source, each offering unique benefits.

Neutron reactor sources, such as the Institut Laue-Langevin (ILL) in Grenoble, France, are currently the most common format of neutron source and are capable of producing the highest average neutron flux, the number of neutrons per second per unit area, for example, the High-Flux Reactor at the ILL is capable of producing a neutron flux of $1.5 \times 10^{15}\,\mathrm{neutrons\,s^{-1}cm^{-2}}$ [5]. A reactor source operates on the principle of nuclear fission, where an atomic nucleus is capable of breaking down into smaller nuclei, overcoming the strong nuclear force. This often involves using uranium enriched with its fissile isotope, ^{235}U, which after the initial absorption of a stray neutron, from a cosmic ray, or spontaneous fission, will undergo fission to release, on average, 2.5 daughter neutrons, an example of a possible uranium fission mechanism is:

$$\mathrm{n} + {}^{235}\mathrm{U} \longrightarrow {}^{236}\mathrm{U} \longrightarrow {}^{134}\mathrm{Xe} + {}^{100}\mathrm{Sr} + 2\,\mathrm{n}.$$

This type of mechanism is the basis for research, and nuclear power, reactors [4]. One of the major drawbacks for reactor neutron sources is the perceived public opinion towards such facilities. Major safety concerns, such as "nuclear meltdown" and the resulting nuclear waste, mean that reactor sources are often unpopular and therefore struggle to obtain funding required for operation.

The other form of neutron source is a spallation source, this is much less controversial as it does not require fissile materials and hence there is no risk of a nuclear disaster. The ISIS Neutron and Muon Source (Oxfordshire, UK) is an example of a spallation source, where high energy protons, $800\,\mathrm{MeV}$ [6], are accelerated towards a tungsten target. When the protons strike the target, they can cause the release of a series of neutrons, the first batch of neutrons are given off with too high an energy to be useful, however, less excited neutrons are given off by secondary emissions. In addition to the public perception benefit, spallation sources also have a technological advantage in the time-of-flight technique. The time-of-flight (ToF) technique is based on the fact that at a spallation source, it is possible to know the time at which the neutron was ejected by the target to a high level of precision and therefore it is possible to measure the time taken for the neutron to reach the instrument. Since the neutron is a particle of a finite mass, m, it is possible to correlate the velocity, v, of the particle with the kinetic energy, E_k,

$$E_k = \frac{mv^2}{2}, \tag{2.2}$$

and with knowledge of the energy of the particle, its wavelength λ, can be determined by the de Broglie relation,

$$E = h\omega = \frac{hv}{\lambda}, \tag{2.3}$$

where, h is Planck's constant and ω is the neutron frequency. Therefore, the wavelength of the neutron is proportional to the inverse of the particle's velocity, and hence the time-of-flight, t_F,

$$\lambda = \frac{h}{mv} = \frac{ht_F}{mL_F}, \tag{2.4}$$

where, L_F is the distance between the target and the instrument. The fact that the neutrons can spread out in the flight from the target means that wavelength-dispersive techniques, where the neutron wavelength is measured rather than the scattering angle, are possible at spallation sources which cannot be carried out at reactor sources. The negative side-effect

of current spallation sources is that they have a lower average flux than reactor sources, however, the building of the European Spallation Source (ESS) will change this as it offers an average flux similar to that of a reactor source, but with the benefits of the spallation technique.

A problem that is inherent for both reactor and spallation sources is that the energy of the neutrons given off is usually too high to be used to study condensed materials, such as soft matter. This means that moderation must be used to reduce the energy of the neutrons passing through the sample. The neutrons which are considered to be optimal for the study of condensed materials are thermal neutrons, named because their energy is approximately that of ambient temperature. Thermal neutrons are achieved by allowing the neutrons to pass through a large volume of moderator material, usually, graphite heavy water (D_2O), methane or H_2, stored at $300\,\mathrm{K}$ before they reach the instrument [4].

2.2 Scattering

The use of scattering techniques to probe soft condensed matter systems is commonplace. In this work, we have focussed on the use of small angle scattering (SAS), reflectometry, and grazing incidence small angle scattering (GiSAS) techniques. These are particularly appropriate for application to soft condensed matter systems due to the length scales capable of being probed being similar to the persistence length of the soft condensed matter systems. The length scale covered for such techniques is from around $1\,\mathrm{nm}$ to $300\,\mathrm{nm}$, as is shown in Figure 2.3. The focus is on the equilibrium structure(s) of a material, and therefore there is no interest in the system dynamics, meaning that exclusively elastic scattering techniques may be used, where there is no energy transfer between the probing radiation and the material. This is in contrast to inelastic scattering where energy transfer occurs; facilitating the measurement of system dynamics, such as the dynamical modes of polymers and lipid bilayers [7, 8].

Both X-ray and neutron scattering techniques are discussed and used in this work. From an experimental viewpoint, there are significant differences between an X-ray scattering and a neutron scattering experiment. However, there is little variation in terms of the data analysis, where the differences are limited to; the nature of the scattering lengths, and the higher background that is present in the neutron scattering experiments.

2.2.1 The scattering vector

The scattering of some probing radiation, by some sample, can be represented as shown in Figure 2.4. Since only elastic scattering is being considered, there will be no change in the frequency of the radiation, $\omega_i = \omega_f$. This means that only the wavevector, $\mathbf{k}$, can change, $\mathbf{k}_i \neq \mathbf{k}_f$. The difference between the incident and final wavevectors is the scattering vector, $\mathbf{q}$, where,

$$\mathbf{q} = \mathbf{k}_i - \mathbf{k}_f. \tag{2.5}$$

The scattering vector strictly has units of m^{-1}, however it is often more practical to use nm^{-1} or Å^{-1}. Throughout this work, units of reciprocal Ångstrom will be wherever possible. Since the frequency of the probing radiation does not change during an elastic scattering event, the wavelength, λ, will also not change, meaning that the moduli of the incident and final

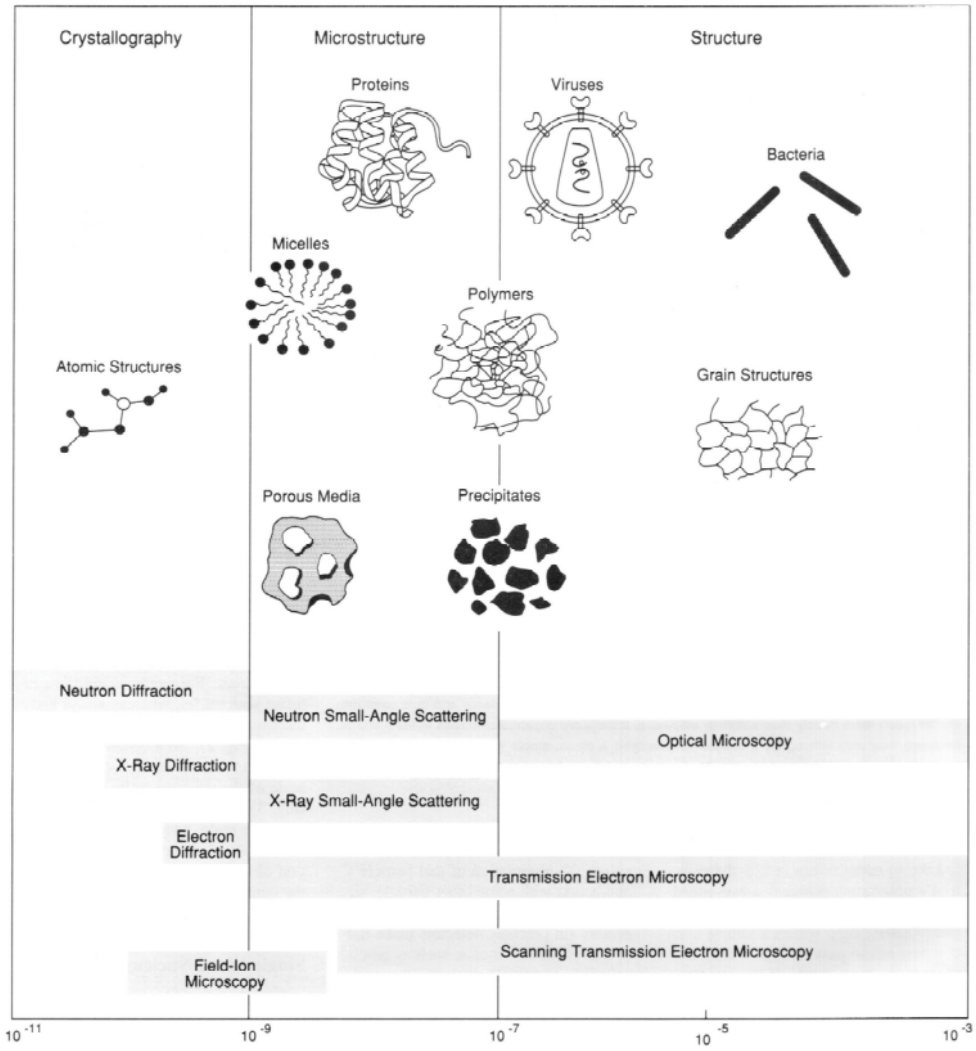


FIGURE 2.3: A representation of how different techniques can be used to probe various length scales. Reproduced, with permission of Oxford University Press©, from Reference [4].

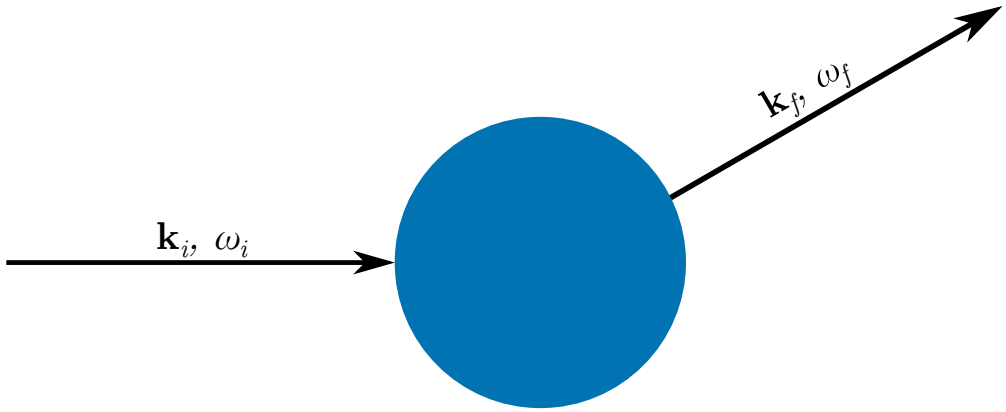


FIGURE 2.4: A schematic of the scattering of some probing radiation by a sample (blue circle). Adapted, with permission of Oxford University Press©, from Reference [4].

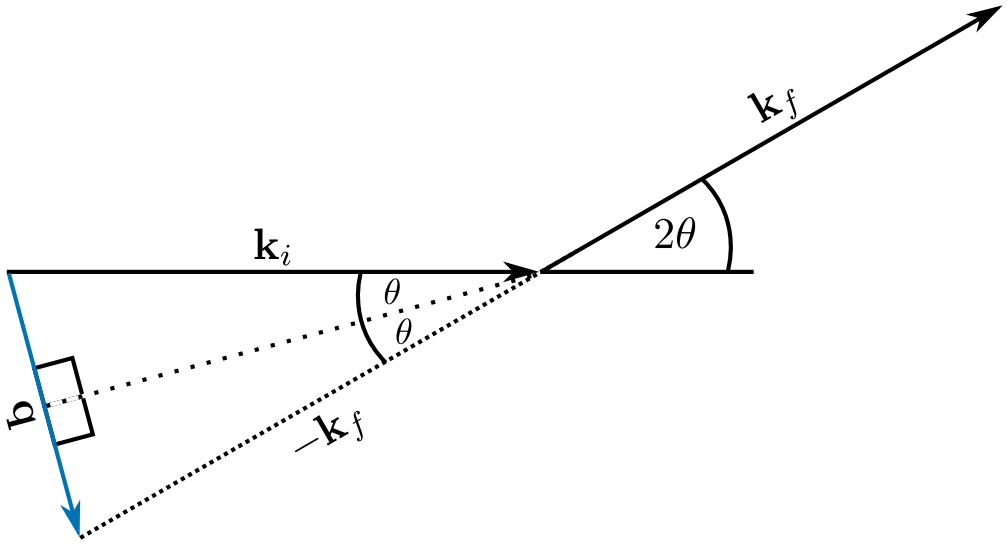


FIGURE 2.5: A vector diagram describing an elastic scattering event. Adapted, with permission of Oxford University Press©, from Reference [4].

wavevectors are,

$$|\mathbf{k}_i| = |\mathbf{k}_f| = \frac{2\pi}{\lambda}. \tag{2.6}$$

This means that only the angle will change during the elastic scattering event. The vector diagram in Figure 2.5 can be used to describe the geometry of an elastic scattering event. From this, and Equation 2.6, the value of q, where $q = |\mathbf{q}|$ can be shown as,

$$q = \frac{4\pi \sin\theta}{\lambda}. \tag{2.7}$$

However, this fails to fully capture the three dimensional nature of the scattering event. Hence, it is necessary to describe the scattering with spherical coordinates, 2θ, and ϕ, such that the incoming and outgoing radiation can be described as,

$$\begin{aligned} \mathbf{k}_i &= \left(0, 0, \frac{2\pi}{\lambda}\right), \\ \mathbf{k}_f &= \frac{2\pi}{\lambda}(\sin 2\theta \cos\phi, \sin 2\theta \sin\phi, \cos 2\theta), \end{aligned} \tag{2.8}$$

where, $|\mathbf{k}_f| = {}^{2\pi}/_{\lambda}$. This allows the scattering vector to be written,

$$\mathbf{q} = \frac{4\pi \sin\theta}{\lambda}(-\cos\theta\cos\phi, -\cos\theta\sin\phi, \sin\theta). \tag{2.9}$$

For an isotropic scattering pattern, it is the magnitude of the scattering vector, q, that is measured. In practical terms, the scattering vector allows for easy comparison of measurements made at different radiation wavelengths.

The basic quantity measured in a scattering experiment is the differential cross section, ${}^{\mathrm{d}\sigma(q)}/_{\mathrm{d}\Omega}$. This is the fraction of particles of probing radiation that is scattered with a particular set of polar coordinates, 2θ and ϕ,

$$\frac{\mathrm{d}\sigma(q)}{\mathrm{d}\Omega} = \frac{R(2\theta, \phi)}{NV\Phi\Delta\Omega}, \tag{2.10}$$

where, $R(2\theta, \phi)$ is the rate of arrival of the scattered particles at the position 2θ, ϕ, V is the illuminated volume of the sample, Φ is incident flux, $\Delta\Omega$ is some small solid angle, and N is the number of scattering particles of interest, in the case of elastically scattered radiation, $N = N\%_{\mathrm{el}}$, where $\%_{\mathrm{el}}$ is the fraction of elastically scattered radiation.

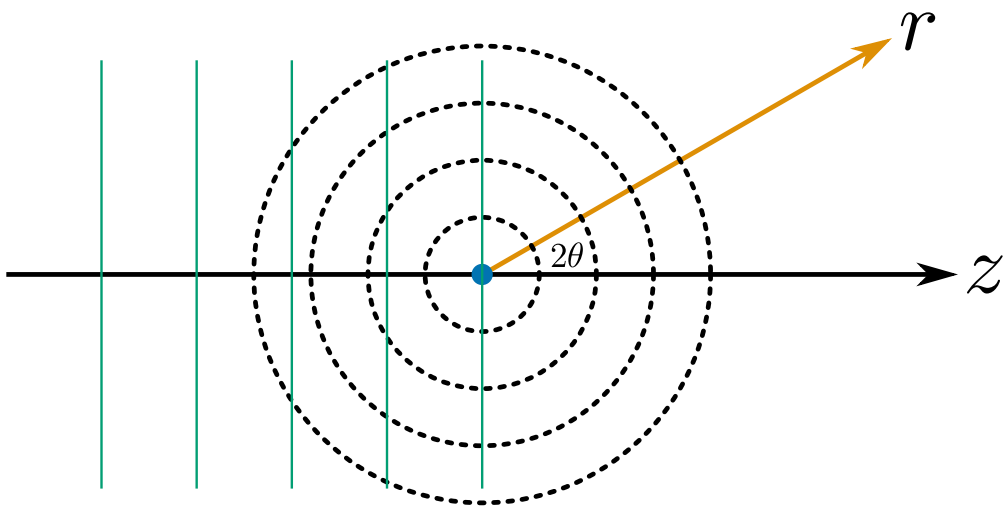


FIGURE 2.6: A schematic of the interaction between a single particle and a wave of probing radiation (green lines). Adapted, with permission of Oxford University Press©, from Reference [4].

2.2.2 Scattering from a single fixed particle

It is possible to describe a steady stream X-ray photons or neutrons of wavelength, λ, travelling through space as follows,

$$\psi_i = \psi_o \exp(\mathbf{i}kz), \tag{2.11}$$

where, z is the direction of travel, and the incident flux is the magnitude of the wave squared, $\Phi = |\psi_o|^2$. This wave then interacts with a single fixed particle elastically, propagating the wave radially outwards (Figure 2.6). This propagation is centred on the atom, therefore the wavevector, $\mathbf{k_f}$ is parallel to the displacement vector, $\mathbf{r}$, and the following holds,

$$\exp(\mathbf{i}\mathbf{k}_f \cdot \mathbf{r}) = \exp(\mathbf{i}kr). \tag{2.12}$$

This final wave is no longer collimated and therefore diminishes with distance, r. Hence the final scattered wave has the form,

$$\psi_f = \psi_o b \frac{\exp(\mathbf{i}kr)}{r}, \tag{2.13}$$

where, b is the scattering length discussed in Section 2.2.9.

2.2.3 Scattering from multiple particles

It is important to consider how the probing radiation would interact with a real system, consisting of many particles. If the incident beam has the form of Equation 2.11, with the wavevector $\mathbf{k}_i = (0, 0, k)$, each particle, j, will contribute the following to the total scattered wave, psi_f, made up of the scattering from all, N, atoms,

$$[\delta\psi_f]_j = \psi_o \exp(\mathbf{i}\mathbf{k}_i \cdot \mathbf{R}_j) b_j \frac{\exp\{\mathbf{i}\mathbf{k}_f \cdot (\mathbf{r} - \mathbf{R}_j)\}}{|\mathbf{r} - \mathbf{R}_j|}, \tag{2.14}$$

where, $\mathbf{R}_j$ is the position of particle j, $\mathbf{r}$ is some arbitrary position, and $\mathbf{k}_f$ is the wavevector of the scattered wave (Figure 2.7). This allows the total scattered wave to be defined as a

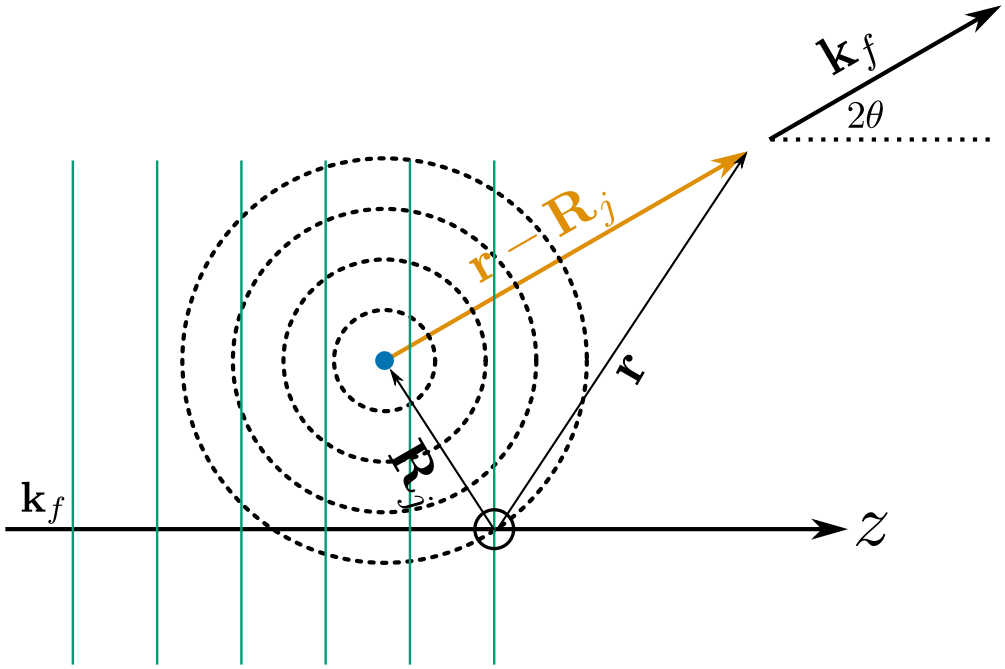


FIGURE 2.7: A schematic of the interaction between a particle, j, at position $\mathbf{R}_j$ and a wave of probing radiation (green lines). Adapted, with permission of Oxford University Press©, from Reference [4].

summation of the contributions from the individual waves,

$$\psi_f = \psi_o \exp\left(\mathbf{i}\mathbf{k}_f \cdot \mathbf{r}\right) \sum_{j=1}^{N} \left\{ b_j \frac{\exp\left(\mathbf{i}\mathbf{q} \cdot \mathbf{R}_j\right)}{|\mathbf{r} - \mathbf{R}_j|} \right\}. \tag{2.15}$$

Equation 2.15 holds true, within the Born approximation, where the scattered wave has no impact on the incident wave and each wave is scattered only once.

The sample-detector distance is usually much larger than the typical particle size, allowing for the following approximation,

$$|\mathbf{r} - \mathbf{R}_j| = |\mathbf{r}| = r. \tag{2.16}$$

This is termed the Fraunhofer, or far-field limit, and allows Equation 2.15 to be simplified,

$$|\psi_f|^2 = \frac{\Phi}{r^2} \left| \sum_{j=1}^{N} b_j \exp\left(\mathbf{i}\mathbf{q} \cdot \mathbf{R}_j\right) \right|^2. \tag{2.17}$$

In the scattering experiment, radiation is deflected elastically into a detector with a small area, δA, with the polar coordinates, 2θ and ϕ, at a rate of R_{el},

$$R_{\text{el}}(2\theta, \phi) = |\psi_f|^2 \delta A = \Phi \delta\Omega \left| \sum_{j=1}^{N} b_j \exp\left(\mathbf{i}\mathbf{q} \cdot \mathbf{R}_j\right) \right|^2, \tag{2.18}$$

where, $\delta\Omega = {}^{\delta A}/_{r^2}$. Therefore, the differential cross section, defined in Equation 2.10 can be related to the scattering from the sample as,

$$\left(\frac{\mathrm{d}\sigma(q)}{\mathrm{d}\Omega}\right)_{\text{el}} = \frac{1}{V} \left| \sum_{j=1}^{N} b_j \exp\left(\mathbf{i}\mathbf{q} \cdot \mathbf{R}_j\right) \right|^2. \tag{2.19}$$

2.2.4 Scattering length density

While it may be helpful to consider the scattering of multiple particles individually, where each particle has a scattering length, b. In practice, due to the low experimental resolution at small angles, it is more common to consider the scattering length density, ρ, of the system,

$$\rho = \frac{1}{V}\sum_{i=0}^{N} b_i, \tag{2.20}$$

where N is the total number of particles in the volume V. A result of this equation is the ability to rewrite Equation 2.19 as,

$$\left(\frac{\mathrm{d}\sigma(q)}{\mathrm{d}\Omega}\right)_{\mathrm{el}} = \frac{1}{V}\left|\iiint_{V} \rho \exp\left(\mathbf{iq}\cdot\mathbf{R}\right)\mathrm{d}^3\mathbf{R}\right|^2. \tag{2.21}$$

This above equation shows that the scattering differential cross-section from some object is related to the scattering length density profile of that object by a Fourier transform.

2.2.5 Model-dependent analysis

All types of scattering patterns can be analysed by one of two methods; model independent and model-dependent. The nature of this work means that it will focus on model-dependent analysis methods, often where the model is derived from some atomistic, or coarse-grained simulation. Model-dependent analysis has significant benefits over model-independent methods, such as improved resolution and more detailed information about the structure. However, the necessity of the inclusion of *a priori* information within model-dependent analysis may act to bias the result. While this is undesirable, these assumptions can, and should, be educated based on the chemical information present, such as the propensity for twin-tailed lipid molecules to form monolayers at an air-water interface [9].

The scattering from the model system is determined, using technique specific methods that are discussed in detail in later sections. This is then compared with the experimental data using some goodness-of-fit metric, the model is then varied to find the best possible model for the data provided using some optimisation algorithm. In order to accurately reproduce the experimental measurement, it is necessary to include some instrumental resolution function, $res(q)$, in the modelling procedure. This is instrument-specific, although it may be approximated by convolving the experimental dataset with some Gaussian smearing function, the modelled intensity can then be determined from,

$$I(q) = res(q) * \frac{\mathrm{d}\sigma(q)}{\mathrm{d}\Omega}, \tag{2.22}$$

where, $\mathrm{d}\sigma(q)/\mathrm{d}\Omega$ is the differential cross-section, a measure of the number of scattering particles hitting a given solid angle of the detector.

The aim of model-dependent analysis is to obtain a model for the system which agrees well with the experimentally measured scattering data while producing something that is chemically, and physically relevant. This means that an optimisation algorithm must be applied to these problems, specific algorithms used in this work are discussed in Section 2.4.3 and 2.4.2.

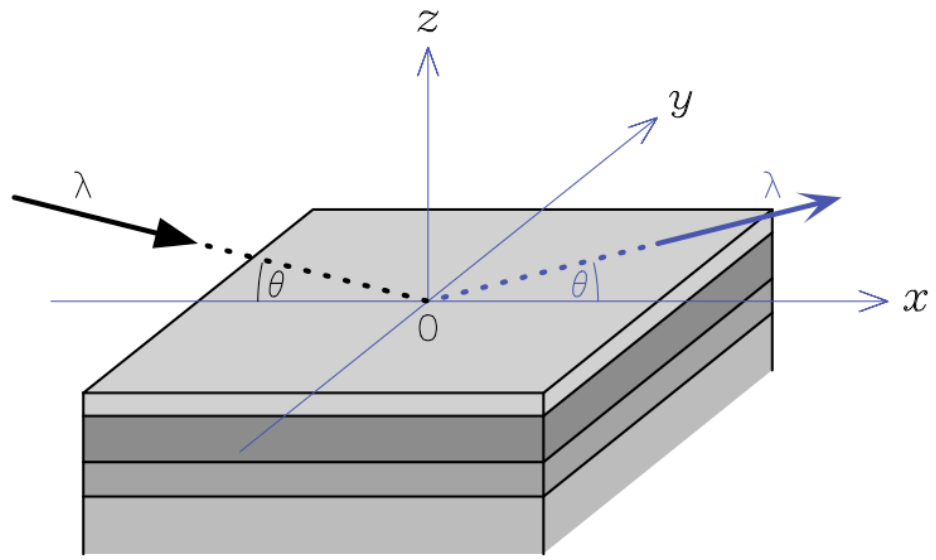


FIGURE 2.8: A schematic of specular reflectometry from a layered sample. Reproduced, with permission of Oxford University Press©, from Reference [4].

2.2.6 Reflectometry

Reflectometry involves the interaction of the probing radiation with some interface, from which the radiation is reflected. The geometry of a reflectometry experiment is shown in Figure 2.8, where the reflectometry instrument is in the horizontal configuration, ideal for the study of liquid interfaces. Reflectometry measurements give information about the structure perpendicular to the interface, the z-dimension in Figure 2.8, and therefore the analysis of reflectometry data is founded on the assumption that the layers will be completely homogenous in the plane of the interface, the xy-plane in Figure 2.8. In reality, since the layers are usually not completely homogeneous, an average is obtained for the area in the radiation beam. A reflectometry instrument operates by measuring the intensity of specular radiation at a series of different angles, θ, or wavelengths, λ. The reflected intensity is defined in terms of q (by Equation 2.7), and is defined as follows,

$$R(q) = \frac{\text{specular reflected radiation at } q}{\text{incident radiation}}. \tag{2.23}$$

It is clear from Equation 2.23 that the value of the measured reflectometry cannot be greater than one, as this would mean that more particles of probing radiation were being reflected than were incident.

Analysis

There are two model-dependent analysis techniques that can be applied to the rationalisation of a reflectometry dataset. The first is the kinematic approach, which can be obtained from Equation 2.19, from the assumption that $q_x = 0$ and $q_y = 0$, as we are only measuring the specular scattering. This approach models the reflectometry as a function of the scattering length density profile in the z-dimension, $\rho(z)$,

$$R(q) \approx \frac{16\pi^2}{q^4}\left|\int_{-\infty}^{+\infty}\frac{\mathrm{d}\rho(z)}{\mathrm{d}z}\exp\left(-\mathbf{i}zq_z\right)\mathrm{d}z\right|^2, \tag{2.24}$$

where, $\mathrm{d}\rho(z)/\mathrm{d}z$ is the first derivative of the scattering length density profile. However, this method has a significant problem, which can be demonstrated by applying Equation 2.24 to the scattering length density profile of a bare silicon substrate, which can be modelled as a Heaviside function (Figure 2.9a),

$$\rho(z) = \begin{cases} 0, & \text{where } z < 0 \\ \rho_{\mathrm{Si}}, & \text{otherwise} \end{cases} \tag{2.25}$$

where, ρ_{Si} is the scattering length density of pure silicon (2.1×10^{-6} Å^{-2} for neutrons). The derivative of a stepwise Heaviside function is a scaled δ-function (Figure 2.9b),

$$\rho'(z) = \rho_{\mathrm{Si}}\delta(z). \tag{2.26}$$

Then, as in Equation 2.24, the Fourier transform of this δ-function is taken,

$$\rho_{\mathrm{Si}} \int_{-\infty}^{+\infty} \delta(z) \exp\left(-\mathbf{i}zq_z\right)\mathrm{d}z = \rho_{\mathrm{Si}} \exp(0) = \rho_{\mathrm{Si}}. \tag{2.27}$$

This means that the reflectometry profile could be calculated from the following relationship,

$$R(q) \approx \frac{16\pi^2\rho_{\mathrm{Si}}^2}{q_z^4}. \tag{2.28}$$

The curve from this relationship is shown in Figure 2.9, where it is clear that the agreement with an experimental profile would be poor as $q \to 0$. It can be seen that for low values of q the calculated reflectometry is greater than 1, which violates the physical constraint imposed with Equation 2.23. This break down of the kinematic approach is due to the assumption present in this approach that the Born approximation (mentioned in Section 2.2.3) will hold. However, in the reflectometry scattering geometry, this is no longer true rendering the kinematic approach invalid.

This breakdown of the kinematic approach has led to the application of the Abelès, or Parratt, model for the reflection of light at a given number of stratified interfaces (also known as dynamical theory) [10, 11]. This method involves considering the system as a layered structure at the interfaces of which the probing radiation can either be reflected or refracted, by some refractive index, n_i. Figure 2.10 shows this process for a system of two layers, where the layer 0 is the air or vacuum above the sample, it is clear to see how the two waves labelled r could interfere constructively or destructively depending on the thickness of layer 1, d. This means that for a single interface, such as that between layers 0 and 1 in Figure 2.10, the reflectometry can be described by the Fresnel equation,

$$R(q) = \left| \frac{n_0 \sin\theta_0 - n_1 \sin\theta_1}{n_0 \sin\theta_0 + n_1 \sin\theta_1} \right|^2. \tag{2.29}$$

Additionally at the point of total reflection, where $\theta_0 = \theta_{\mathrm{c}}$, the critical angle, there will be no transmitted wave so,

$$n_1 \sin\theta_1 = 0, \tag{2.30}$$

and therefore the reflected radiation will never be greater than 1, while the critical angle can be defined as,

$$\cos^2\theta_{\mathrm{c}} = \frac{n_1^2}{n_0^2}. \tag{2.31}$$

This is the angle below which a reflectometry profile will be measured.

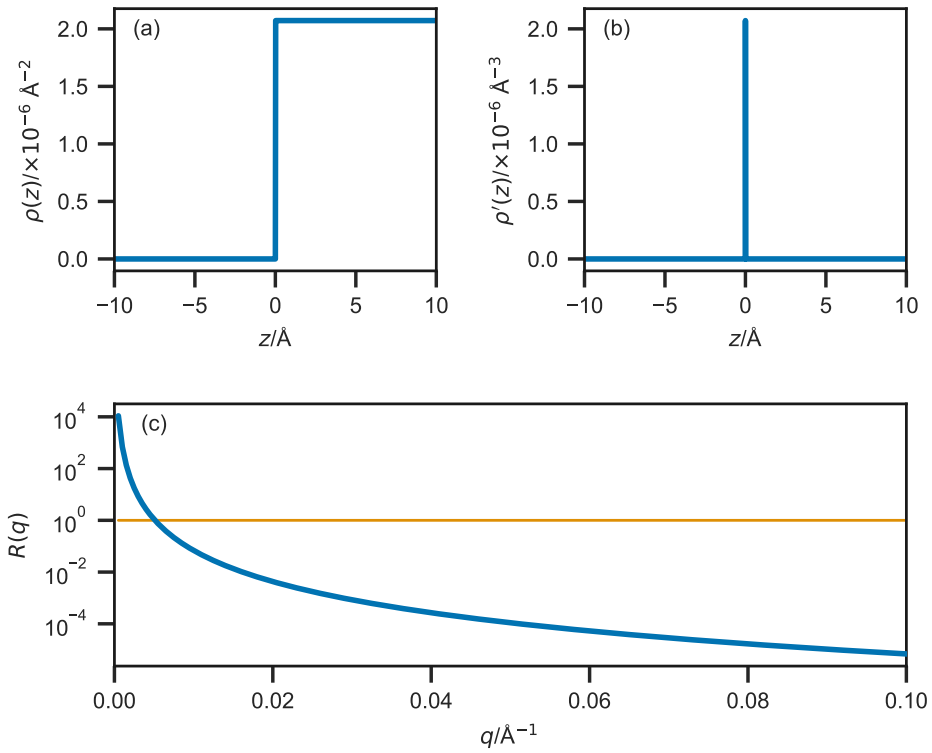


FIGURE 2.9: A graphical representation of the kinematic approach; (a) the Heaviside function describing the scattering length density profile of a bare silicon substrate, (b) the δ-function arising from the first derivative of the function in (a), and (c) the reflectometry profile resulting from Equation 2.24, where the orange line at $R = 1$ identifies the break down between experimental and theory in the kinematic approach. Adapted, with permission of Oxford University Press©, from Reference [4].

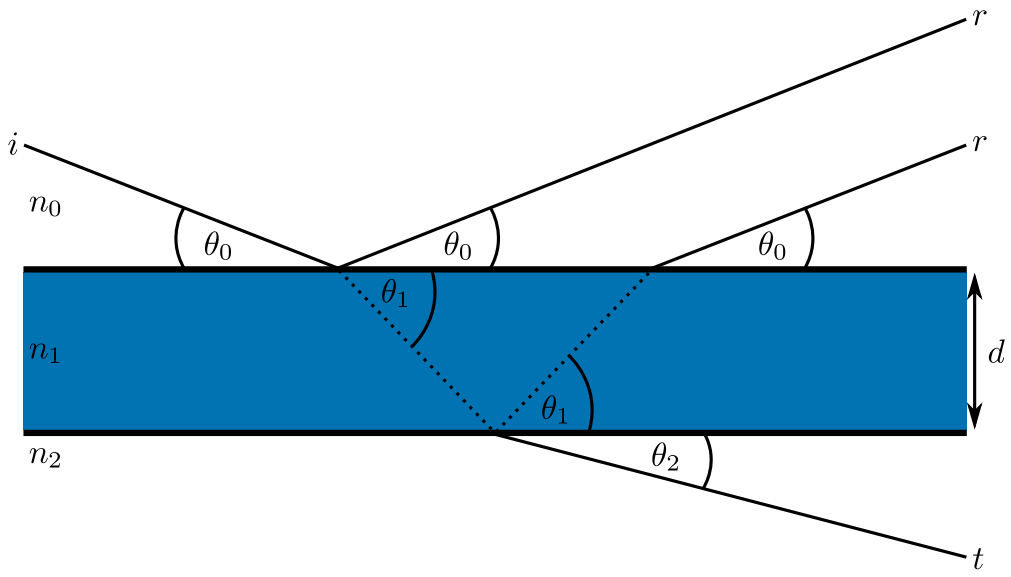


FIGURE 2.10: A schematic diagram showing the reflected (r) and transmitted (t) waves when an incident (i) wave enters an interface of thickness d, where the refractive indices of each layer are n_0, n_1, and n_2. Adapted from [12], with permission from Elsevier.

The above method can then be generalised to a structure of an arbitrary number of layers, as shown in Code Block 2.1. For each value of q for which the reflectometry is to be calculated, the system is considered in terms of n_{max} layers. The incident radiation beam will be refracted by each of the layers, giving wavevectors values for each layer, k_n,

$$k_n = \sqrt{k_0^2 + 4\pi(\rho_n - \rho_0)}, \tag{2.32}$$

where, $k_0 = q/2$. The Fresnel equation coefficient between layers n and $n+1$, $r_{n,n+1}$ can then be found along with the phase factor, β_n, which is dependent on the thickness of the layer, d_n,

$$r_{n,n+1} = \frac{k_n - k_{n+1}}{k_n + k_{n+1}}, \tag{2.33}$$

$$\beta_n = k_n d_n. \tag{2.34}$$

The means that a matrix can be evaluated for each layer, M_n,

$$M_n = \begin{bmatrix} \exp \beta_n & r_{n,n+1} \exp -\beta_n \\ r_{n,n+1} \exp \beta_n & \exp -\beta_n \end{bmatrix} \tag{2.35}$$

The resultant matrix, B_n, is then found as a product of the matrix from each layer,

$$B = \prod_{n=0}^{n_{\text{max}}} M_n, \tag{2.36}$$

and from this the reflected intensity at the given value of q can be found,

$$R(q_z) = \frac{B_{1,2}}{B_{1,1}}. \tag{2.37}$$

This algorithm models the layers as perfectly flat layers, which will not be strictly true in the case of soft matter systems such as a bilayer. This resulted in the correction term being added to Equation 2.33 to account for the roughness of the layers. This adapts Equation 2.33 to the form,

$$r_{n,n+1} = \frac{k_n - k_{n+1}}{k_n + k_{n+1}} \exp\left(-2k_n k_{n+1} \sigma_{n,n+1}^2\right), \tag{2.38}$$

where, $\sigma_{n,n+1}$ is the interfacial roughness between layers n and $n+1$. This has the effect of Gaussian broadening the layers into each other, as a result. Code Block 2.1 is currently implemented in a variety of reflectometry modelling software packages, such as `refnx`, MOTOFIT, RasCAL, and Aurore [13–17]. Applying this method to the scattering length density profile shown in Figure 2.9 gives the reflectometry profile shown with the dashed green line in Figure 2.11.

2.2.7 Small angle scattering

Equation 2.21 identified that the scattering differential cross-section for some object was related to the scattering length density by a Fourier transform, which is shown graphically in Figure 2.12. This figure shows that there is a reciprocal relationship between the size of the object and the scattered intensity, decaying significantly up to values of $2\pi/d_x$, where d_x is the size of the object. This means that in order to probe the large-scale structural features that are of interest in the study of soft materials, it is necessary to consider small values of q. When considering the nature of q in Equation 2.7, it is clear that such experiments would

CODE BLOCK 2.1: An example code block for the Abelès method for the calculation of reflectometry, adapted from Reference [13].

```
import numpy as np


def abeles(q_values, sld, d):
    """
    Calculates the reflectometry from a set of layers of a given
    scattering length density.

    Parameters
    ----------
    q_values: float, array-like
        The q-vector values over which the reflectometry is to be
        calculated.
    sld: float, array-like
        An array of scattering length densities of length N, where
        N is the number of layers present.
    d: float, array-like
        An array of thicknesses of length N, where N is the number
        of layers present.

    Returns
    -------
    float, array-like
        The reflected intensity over the given q-vector values.

    """
    R = np.zeros_like(q_values)
    kn = np.sqrt(
        q_values[:, np.newaxis] ** 2.0 / 4.0 - 4.0 * np.pi * sld
    )
    B = np.zeros((2, 2, q_values.size))
    B[0, 0, :] = 1
    B[1, 1, :] = 1
    k = kn[:, 0]
    nmax = sld.size
    for n in range(1, nmax):
        kn1 = kn[:, n]
        r = (k - kn1) / (k + kn1)
        betan = k * d[n]
        if n > 0:
            Mn = np.array(
                [
                    [np.exp(betan * 1j), r * np.exp(betan * 1j)],
                    [r * np.exp(-betan * 1j), np.exp(-betan * 1j)],
                ]
            )
        else:
            Mn = np.array([[1, r], [r, 1]])
        p0 = B[0, 0, :] * Mn[0, 0, :] + B[1, 0, :] * Mn[0, 1, :]
        p1 = B[0, 0, :] * Mn[1, 0, :] + B[1, 0, :] * Mn[1, 1, :]
        B[0, 0, :] = p0
        B[1, 0, :] = p1
        p0 = B[0, 1, :] * Mn[0, 0, :] + B[1, 1, :] * Mn[0, 1, :]
        p1 = B[0, 1, :] * Mn[1, 0, :] + B[1, 1, :] * Mn[1, 1, :]
        B[0, 1, :] = p0
        B[1, 1, :] = p1
        k = kn1
    R = (B[0, 1, :] * np.conj(B[0, 1, :])) / (
        B[0, 0, :] * np.conj(B[0, 0, :])
    )
    R[np.where(np.isnan(R))] = 1.0
    return np.real(R)
```

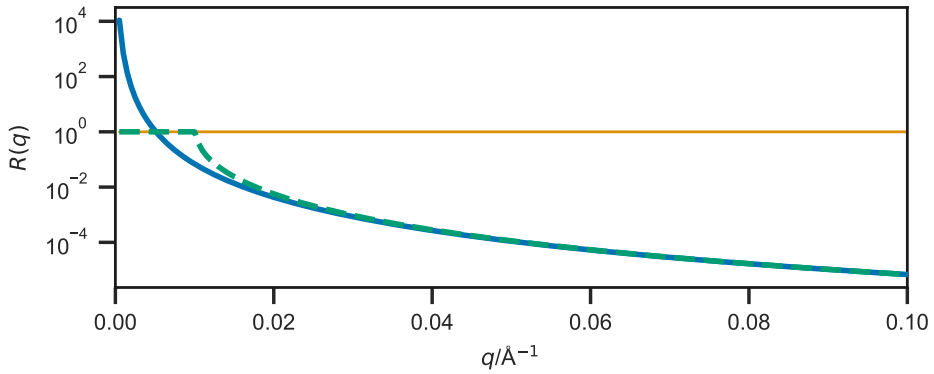


FIGURE 2.11: A comparison of the kinematic approach (blue solid line), and the dynamical approach (green dashed line), to determine the reflected intensity from the material with the scattering length density profile given in Figure 2.9(a). It is clear that at low q, there is a noticable deviation between the two.

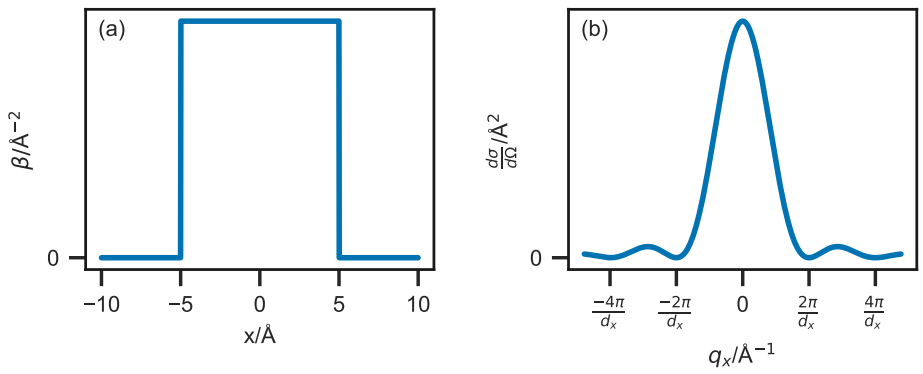


FIGURE 2.12: The effect of a Fourier transform (a) the scattering length density profile for some object with a width of 10 Å, (b) the Fourier transform of this object showing the minima in the differential cross section at values of $^{2n\pi}/_{10}$, where n is some integer.

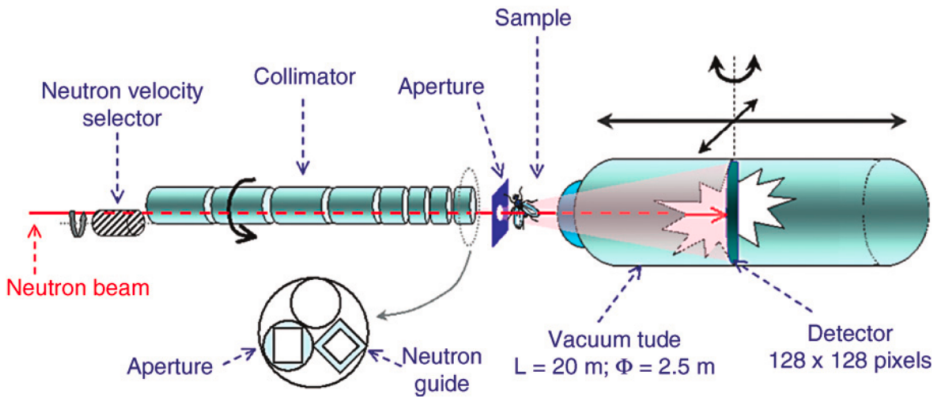


FIGURE 2.13: A schematic of the D22 instrument of the ILL, from Reference [20]. Reprinted/adapted by permission from Springer Nature Customer Service Centre GmbH: Springer Nature Small-Angle Neutron Scattering and Applications in Soft Condensed Matter by I. Grillo© (2008).

benefit from small values of θ and large values of λ. Hence, the application of small angle scattering (SAS).

A SAS experiment generally involves some sample being placed in the path of the probing radiation; the scattering pattern that results from this transmission is measured at some distance, as is shown in Figure 2.13 for the D22 SANS instrument of the ILL. SAS instruments are usually very large, due to the large post sample flight path that is necessary to reach the small angles being measured; a longer flight path allows more space for angular divergence. Transmission SAS can provide information about the size, shape and orientation of the sample's components [18]. The range in q that is typically covered by a SAS instrument is usually around 2×10^{-3} to $0.5\,\text{Å}^{-1}$, which corresponds to 10 to 3000 Å in real-space. The neutron or X-ray detector of a SAS instrument is often two-dimensional, meaning that for an isotropic scattering profile, the detector image is radially averaged to give an $I(q)$ scattering profile. It is possible to increase the q-range of a SAS instrument through the introduction of wide-q detector banks close to the sample or small-q detector banks further away. This allows the SANS2D instrument, at the ISIS Neutron and Muon Source, to have a total range from 2×10^{-3} to $2\,\text{Å}^{-1}$ [19]. Furthermore, the SANS2D instrument may leverage the time-of-flight method discussed in Section 2.1.2 to allow for a much shorter post sample flight path than is present at D22.

Analysis

A radially averaged SAS pattern can be considered as consisting of two sections that arise from the form and structure factors for the scattering species. The form factor gives information about the average shape of the scattering particle, while the structure factor is a measure of the interaction present between the objects. It is often possible to control the presence of the structure factor by changing the concentration of the sample, eventually, the concentration will be so low that all interparticle interaction is screened by the solvent [21]. This method is frequency applied in BioSAXS applications, where the interactions between the protein molecules are of less interest than the overall structure of the complex. However, with micelles, it is not always possible to remove the structure factor, as the critical micelle concentration may be higher than the minimum concentration at which the structure factor

is present. It is possible to deconvolute the structure and form factors for a micellar solution by studying different concentrations, assuming that the form of the micelle is concentration independent, over the measured concentration range.

The rigorous, model-independent method for the analysis of SAS involves taking the inverse Fourier transform of the scattering, to give an auto-correlation function of the average particle in the system, which following a deconvolution procedure will resolve the radially averaged scattering length density profile. However, this is often cumbersome and has a low information density, when compared to model-dependent techniques. Additionally, if the experimental data lacks information at wide enough q to cover all features of the sample artefacts may be present in the inverse Fourier transform of the scattering.

There are two common and straight-forward analysis procedures that can be used to give an understanding of the scattering species. The first is the Guinier approximation, which is used in the determination of the radius of gyration, R_g, of the scattering species, at "infinite dilution". This scattering law is only valid at very small values of q, where $q < R_g^{-1}$ [4],

$$\ln[I(q)] = \ln[I(0)] - \left(\frac{R_g^2}{3}\right)q^2. \tag{2.39}$$

This relationship allows the radius of gyration to be found by plotting the scattering profile transformed into $\ln[I(q)]$ vs. q^2, and evaluating the gradient at low q. The Guinier plot for the scattering from a sphere with a radius of 20 Å is shown in Figure 2.14, where the radius of gyration correlates with the radius of the sphere, R, as follows,

$$R_g = \sqrt{\frac{3}{5}}R. \tag{2.40}$$

The Guinier analysis is very common in the study of proteins by SAS, as it allows for the determination of the protein size in the native, solution phase [22]. Another common analysis of SAS data comes in the form of Porod's law, which states that for large values of q, the scattering intensity becomes proportional to Sq^{-4}, where S is the surface area of the sample. This means that by plotting $I(q)q^4$ vs. q and extrapolating to $q \to \infty$, it is possible to determine the external surface area of the system [18]. Using the surface area it is then possible to qualitatively determine the 'roughness' of the system.

In the calculation of a SAS pattern, both the structure and form factors will contribute. Therefore, when we model the pattern, the differential cross-section has the following form, when the system is centrosymmetric.

$$\frac{\mathrm{d}\sigma(q)}{\mathrm{d}\Omega} = N_\rho \Delta\rho^2 V_p^2 P(q) S(q), \tag{2.41}$$

where N_ρ is the number density of the particles, $\Delta\rho$ is the difference in scattering length density between the particles and the solvent, V_p is the particle volume, $P(q)$ is the particle form factor, and $S(q)$ is the structure factor [23]. Therefore, it is necessary to understand how the function of the form factor and structure factor come to be.

The most common method for the modelling of the form-factor is by using very coarse shapes; such as spheres, cylinders, or ellipses. This involves the evaluation of analytical or quasi-analytical solutions for the scattering, which have been derived for many common shapes. The solution for a sphere was solved in the early 19th century by Lord Rayleigh [23].

$$P(q) = \left\{ \frac{3\{\sin(qR) - qR\cos(qR)\}}{(qR)^3} \right\}^2, \tag{2.42}$$

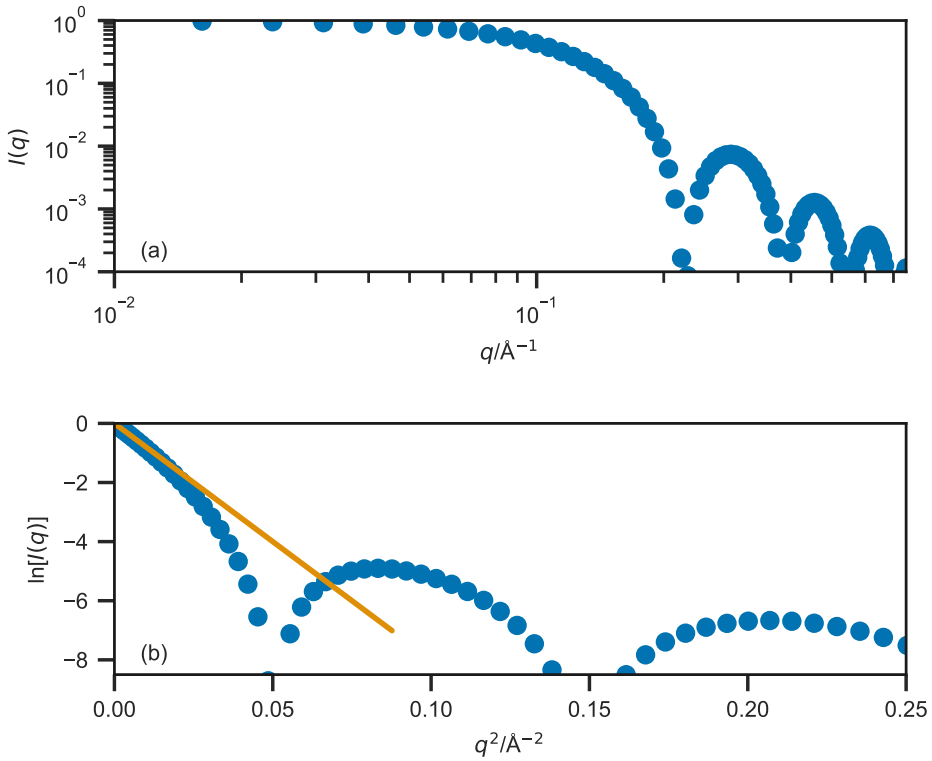


FIGURE 2.14: The Guinier plot, (a) the ideal scattering profile from a sphere of radius 20 Å, (b) the associated Guinier plot, with a straight line (orange) at low-q showing the radius of gyration to be $\sim$15.5 Å.

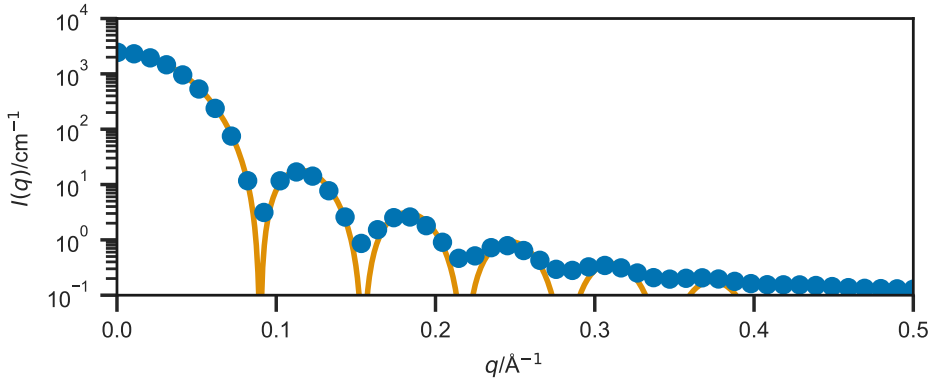


FIGURE 2.15: The SANS profile of a micelle of C_{16}TAB with radius (50 ± 3) Å (circles, generated using SASView [24], with instrumental smearing) compared with a curve of Equation 2.42, where $R = 50$ Å (solid line).

where R is the radius of the sphere. A comparison between a possible experimental scattering pattern and the scattering generated from Equation 2.42 is shown in Figure 2.15. Analytical form factors exist for a wide variety of shapes; these can be found in software such as SASView and SASFit [24, 25].

The structure factor accounts for the scattering interference that arises from the interaction of different particles. This is modelled using expressions which depend on the nature of the scattering particles; hard-sphere, sticky hard-spheres, screened Coulomb, etc. Structure factor expressions are generated as solutions to the Ornstein-Zernike Equation [26]. The most relevant structure factor in terms of micelle modelling is probably the Hayter-Penfold Mean Spherical Approximation [27], this is where the micelles are modelled as like-charged, soft spheres and is valid for dilute solutions. Again, a whole range of these structure factor functions are built into the SASView package [24].

In order to evaluate the scattering profile from an atomic, or coarse-grained system, it is possible to use the Debye equation [28]. This is an analysis relation for the determination of the scattering profile based on the atomic positions, $\mathbf{r}_i$,

$$I(q) = \sum_i^N \sum_j^N b_i b_j \frac{\sin\left(q|\mathbf{r}_i - \mathbf{r}_j|\right)}{q|\mathbf{r}_i - \mathbf{r}_j|}, \tag{2.43}$$

where, N is the number of particles, q is the scattering vector, and b_i and b_j are the scattering lengths of particles i and j respectively, for a coarse-grained particle this can be obtained from summing the scattering lengths of the constituent atoms. The Debye equation is powerful, however, it is not intrinsically parallelisable and as a result scales as $\mathcal{O}(N^2)$. In order to improve the efficiency of the calculation of the scattering profile, a variety of methods have been developed that offer a sufficiently accurate approximation [29, 30]. The Golden Vector method, developed by Watson and Curtis [30] scales as $\mathcal{O}(Nn)$, where n is the number of scattering vectors used in the calculation. In this method, the scattering amplitude is calculated numerically for a single $\mathbf{q}$-vector,

$$I(\mathbf{q}) = \left[\sum_i^N b_i \cos\left(\mathbf{q}\cdot\mathbf{r}_i\right)\right]^2 + \left[\sum_i^N b_i \sin\left(\mathbf{q}\cdot\mathbf{r}_i\right)\right]^2. \tag{2.44}$$

This is carried out for n scattering vectors that are selected in an orientationally averaged fashion from a quasi-uniform lattice on a sphere. This was first developed such that n is a number from the Fibonacci sequence [29], however for the Golden Vector method, n may be any positive integer. This leads to the scattering vectors being calculated as,

$$\begin{aligned} q_x^{(k)} &= q\cos\left[\sin^{-1}\left(\frac{2k}{n}\right)\right]\cos\left(\frac{2\pi k}{\Phi}\right), \\ q_y^{(k)} &= q\cos\left[\sin^{-1}\left(\frac{2k}{n}\right)\right]\sin\left(\frac{2\pi k}{\Phi}\right), \\ q_z^{(k)} &= \frac{2kq}{n}, \end{aligned} \tag{2.45}$$

where, k runs from $-(n-1)/2, \ldots, 0, \ldots, (n-1)/2$ and Φ is the golden ratio,

$$\Phi = \frac{1+\sqrt{5}}{2}. \tag{2.46}$$

The approximate orientationally averaged scattering is then found as an average of the scattering from each of ther individual q-vectors,

$$I(q) = \frac{1}{n}\left\{\sum_{k=(1-n)/2}^{(n-1)/2} I[\mathbf{q}^{(k)}]\right\}. \tag{2.47}$$

The accuracy of this calculation increases with n, however good agreement between experiment and simulation has been shown for $n < 100$ even for highly anisotropic systems [30].

2.2.8 Grazing incidence small angle scattering

While reflectometry is the study of specular reflections from a layer structure, grazing incidence small angle scattering (GISAS) is the study of the off-specular reflections, those where the scattering vector has x- and y-components in addition to the z-component. This off-specular scattering is that which arises from the heterogeneity present at the interface. This means that GISAS is ideal for the study of material with nanostructure in the plane of the interface, such as organic semiconducting materials or polymer-grafted nanoparticles at an interface [31, 32]. The geometry of a GISAS experiment on a thin film material is shown in Figure 2.16, which also shows a typical GISAS pattern.

In terms of GISAS instrumentation, it can be considered as an amalgamation of a reflectometry instrument and a small angle scattering instrument. The sample environment is that from a reflectometry instrument, while the two-dimensional detector required is that commonly associated with small angle scattering. Commonly, GISAS is found as an additional capability of another instrument, such as the grazing incidence small angle neutron scattering (GISANS) capability of D22 at the ILL or the GISAS setup at I07 of the Diamond Light Source [34, 35]. However, as GISAS has grown in popularity as a technique, GISAS focused instruments have been installed, such as MiNaXS at DESY in Hamburg and Beamline 7.3.3 at the ALS in San Francisco [36, 37].

Analysis

The analysis process for a GISAS pattern is substantially more complex than for either the SAS or reflectometry counterparts. This is due to the fact that for SAS, the Born approximation is conserved, and only a single scattering event is considered. While for reflectometry,

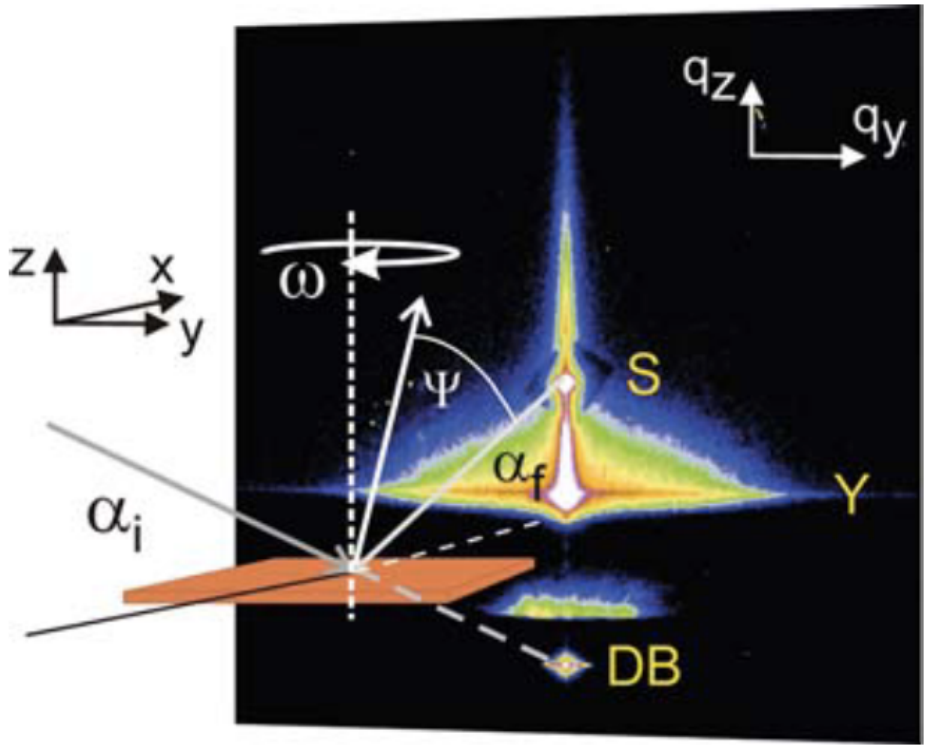


FIGURE 2.16: A schematic of the geometry of a GISAS experiment, where α_i is the incident angle, α_f is the exit angle, and ϕ is the out-of-plane angle, from Reference [33]. Reprinted/adapted by permission from Springer Nature Customer Service Centre GmbH: Springer Nature A Basic Introduction to Grazing Incidence Small-Angle X-Ray Scattering by P. Müller-Buschbaum© (2009).

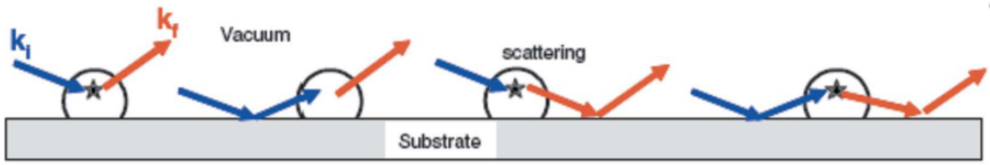


FIGURE 2.17: The four possible way in which some probing radiation may interact with a sample within the DWBA; scattering, reflection-then-scattering, scattering-then-reflection, and reflection-then-scattering-then-reflection, from Reference [38].

there is a well defined theoretical background that enables the analysis where the Born approximation is not valid. This is not the case for GISAS since for off-specular reflectometry to occur, the probing radiation will undergo both scattering and reflectometry events simultaneously. In order to account for this, generally, the distorted wave Born approximation (DWBA) is invoked. This is an extension of the Born approximation, wherein there are four possible ways in which the probing radiation may interact with the material, these are shown in Figure 2.17.

One of the most common methods for the analysis of a GISAS pattern is Bragg peak analysis. This is where Bragg peaks of crystallinity in the GISAS pattern are selected and analysed in terms of Bragg's and Snell's laws to account for the effects of the surface and substrate, this allows the dependence of q_z to be determined [39, 40]. While useful, this analysis approach is limited to systems with high crystallinity, such as stacked lamellae [41]. This method cannot be easily applied to understand less organised arrangements in soft matter systems, as these give rise to broad, smeared peaks with no clear centre.

Another simplified method for the analysis of a GISAS pattern involves the effective surface approximation. This is where, from the two-dimensional GISAS pattern, selected cuts are taken and analysed only as a function of q_y. This can be used to probe depth-dependent properties, as taking such a cut effectively considered the lateral structure at a given depth, rather than both the lateral structure and that normal to the interface. For incident angles $\alpha_i \gg \alpha_c$ and at constant q_z, the scattering intensity in the DWBA simplifies and the differential cross section is given by [42],

$$\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega} = \frac{A\pi^2}{\lambda^4}\left[(1-n^2)T_iT_f\right]^2 F(\mathbf{q}), \tag{2.48}$$

where A is the iluminated surface area, $T_{i,f}$ are the Fresnel transmission factors for incidence and reflectance, and $F(\mathbf{q})$ is the diffuse-scattering factor, which is dependent on the form and structure factors of the lateral structure.

Currently, there exists a handful of software packages that are designed to aid in the structural analysis by GISAS. These include, but are not limited to, IsGISAXS, BornAgain, and HipGISAXS [43–45]. These software packages generally implement analytical solutions to the DWBA by considering different form and structure factors for some system. In order to have some understanding of the structure that should be fitted to the experimental data, it is often necessary to perform electron microscopy measurements to educate the analysis process.

2.2.9 Contrast variation

The scattering profile generated by the interaction of some system with radiation depends on three factors:

- the spatial arrangement of the atoms in the system,
- the instrument being used to measure the pattern; instrumental resolution function, and
- the interaction between the radiation and the matter under investigation.

This final factor is perhaps better known as the 'scattering contrast', this is an extremely important factor in the study of soft matter, particularly when the probing radiation is the neutron. The scattering contrast makes it possible to select individual components of the system and investigate their structural properties [46]. The differential cross-section, $\mathrm{d}\sigma/\mathrm{d}\Omega$ of a point scatterer, as shown in Equation 2.19, varies only with respect to the scattering length of the species, b,

$$\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega} \propto b^2. \tag{2.49}$$

However, as discussed in Section 2.2.4, it is often easier to use the scattering length density, ρ.

When an X-ray interacts with an atom, it is scattered by the interaction with the electrons, this is due to the X-ray being a form of electromagnetic radiation. Furthermore, it means that the scattering length of an atom by an X-ray is directly proportional to the number of electrons in the atom, so it is therefore difficult to discern between the scattering from a carbon atom (6 electrons) and a nitrogen atom (7 electrons), furthermore the scattering from hydrogen atoms is practically non-existent.

The scattering length a neutron by an atom varies unsystematically with respect to the atomic number of a species, this is shown in Figure 2.18. Furthermore to the apparently random variation with changes in atomic number, there is also significant variation with mass number, e.g. between isotopes of the same atom. There is also dependence due to the magnetic state of the atom, however, this is normally unimportant for soft matter. The scattering lengths differ with the nuclear spin energy level, this leads to an average scattering length, $\langle b \rangle$, for isotopes where the nuclear spin is non-zero ($S \neq 0$). There are two forms of scattering, coherent and incoherent, for which the scattering cross-sections, σ, are determined by,

$$\begin{aligned} \sigma_{\mathrm{coh}} &= 4\pi \langle b \rangle^2 \\ \sigma_{\mathrm{incoh}} &= 4\pi (\langle b^2 \rangle - \langle b \rangle^2) \end{aligned} \tag{2.50}$$

The coherent scattering is the scattering from nuclei that all have the same value of b, and leads to the important scattering pattern. Whereas, the incoherent scattering is caused by the 'disorder' between the isotopes, and is the cause of the background present in the measurement. Examples of these scattering cross-sections for nuclei relevant to soft matter are shown in Table 2.2. It can be seen that the incoherent scattering from the ^{1}H nuclei is more than forty times higher than the coherent scattering. This leads to a large, intrusive background present in the scattering pattern of hydrogenous samples.

The difference between the scattering of ^{1}H and ^{2}H, evident in Table 2.2, can lead to a very useful technique in soft matter scattering, known as contrast variation. The idea of contrast variation is based on the substitution of one isotope of an atom for another, while not introducing significant change to the properties of the material. Traditionally the benefit of this

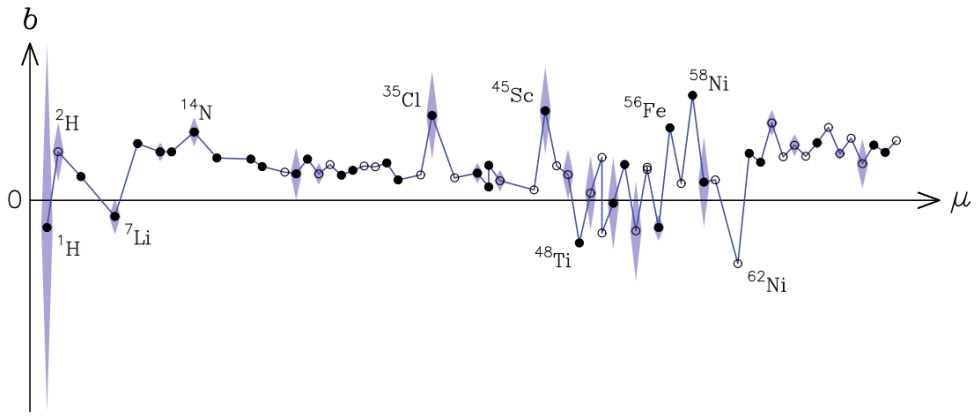
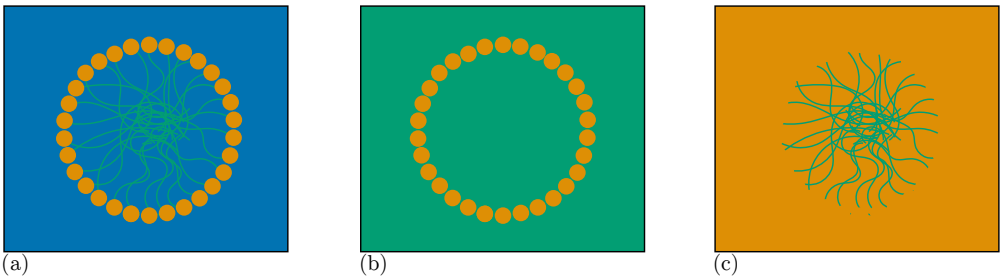


FIGURE 2.18: The variation of the average neutron scattering length, $\langle b \rangle$ (circles), with atomic mass, μ. The standard deviation, Δb, is indicated with the shaded regions. Reproduced, with permission of Oxford University Press©, from Reference [4].

TABLE 2.2: Examples of coherent and incoherent scattering cross-sections, from Reference [46].

Isotope	S	$\sigma_{\text{coh}}/10^{-28}\,\text{m}$	$\sigma_{\text{incoh}}/10^{-28}\,\text{m}$
^{1}H	$^{1}/_{2}$	1.8	79.7
^{2}H	1	5.6	2.0
^{12}C	0	5.6	–
^{14}N	1	11.6	0.3
^{16}O	0	4.2	–



(a) (b) (c)

FIGURE 2.19: The effect of varying the scattering length density of the solvent in a micelle system, (a) the system in a pure solvent, (b) the solvent is contrast matched to the surfactant tails, and (c) the solvent is contrast matched to the surfactant heads.

came in terms of contrast matching out a part of the system to reduce the dimensionality of the problem for analysis. For example, by matching the solvent scattering length density to that of the tails of the surfactants at the centre of a micelle there would only be scattered from the heads, and conversely, there would only be scattered from the tails if the solvent had the same scattering length density as the head groups. This means that the problem becomes more straightforward as there are fewer variable parameters when fitting the data. This idea is represented graphically in Figure 2.19. The technique of contrast variation may also be used in terms of data analysis. By increasing the number of data sets corresponding to a single model at different contrasts, the solution for the true structure of the model from the scattering data becomes more robust. This is due to the fact that each different contrast can be considered as an independent measurement of the same system, and hence each set of scattering data can be used within the data analysis procedure to obtain the best global agreement to the experiment. This co-refinement of multiple experiments can, under the right conditions, be used to simultaneously consider both neutron and X-ray datasets [14].

There is also the possibility of using contrast variation when the probing radiation is the X-ray, through the use of anomalous scattering. This is where different wavelengths of radiation give different scattering when the wavelengths are on opposite sides of an X-ray absorption edge. This is not frequently used for soft matter species, as the X-ray absorption edges for elements common in soft matter (H, C, N, O, etc.) are at very low X-ray energies so generally outside of the accessible range of a standard SAXS instrument [46].

2.3 Classical simulation

While the currently applied traditional methods for the analysis of experimental scattering data discussed previously are popular. There is growing interest in the use of multi-modal analysis methods that leverage classical simulation to assist in the analysis of scattering data [47–50]. This would involve the simulation of the chemical system in order to educate the analysis of the experimental data. These systems, especially when the materials being simulated are soft in nature, are often highly complex and typically cover large length scales. Classical simulation, particularly in combination with coarse-grained potential models, can feasibly enable the simulation of these methods.

In order to simulate a real chemical system, it is necessary to model the electrons of the molecules and their interactions. This is usually achieved using quantum mechanical calculations, where the energy of the system is calculated by finding some approximate solution to the Schrödinger equation. However, quantum mechanical calculations are very computationally expensive and are realistically limited to hundreds of atoms. In order to simulate

a soft matter system such as a lipid monolayer or polymer nanoparticles, it is necessary to simplify the calculation being performed. This leads to classical simulation, where mathematical functions are used to determine the potential energy of the system. Classical simulations are used substantially in this work, in terms of both molecular dynamics simulations and energy minimisation methods (see Section 2.4). Therefore, it is necessary to introduce the underlying theory on which this method is defined.

2.3.1 Potential models

Potential modelling is a more computationally efficient method for the calculation of the potential energy of a chemical system. A potential model consists of a series of mathematical functions that depend on the atomic positions, $\mathbf{r}$. Each of the functions represents the potential energy of a different interaction for a given atom. Broadly, these interactions can be split into bonded and non-bonded, such that the total energy may be described as follows,

$$E_{\text{total}}(r) = E_{\text{bonded}}(r) + E_{\text{non-bonded}}(r) \tag{2.51}$$

The total potential energy is then the sum of the potential energy for each of the individual atoms.

The bonded terms are used to describe different aspects of chemical bonds. These typically consist of bond stretches, angle bends and dihedral torsions; within the OPLS2005 potential model [51], these interactions have the following mathematical form,

$$\begin{aligned} E_{\text{bonded}}(b,\theta,\phi) = &\sum_{\text{bonds}} K_b(b-b_0)^2 + \sum_{\text{angles}} K_\theta(\theta-\theta_0)^2 \\ &+ \sum_{\text{dihedrals}} \frac{1}{2}\big\{A_1[1+\cos(\phi)] + A_2(1-\cos(2\phi)] + A_3(1+\cos(3\phi)]\big\}, \end{aligned} \tag{2.52}$$

where, K_b and b_0, K_θ, θ_0, and A_1, A_2, and A_3 are interaction dependent parameters for the bonds, angles, and dihedrals respectively, while b, θ, and ϕ are the bond lengths, the size of the angles, and the size of the dihedrals that depend on the atom positions. It can be seen that both the bond stretch and angle bend have harmonic functions, whereas the dihedral consists of a more complex multiple cosine functions. The values of the interaction dependent parameters are determined as outlined in Section 2.3.2.

The non-bonded terms are a series of functions that describe the potential energy of intermolecular interactions, such as electrostatics and London dispersion forces. The potential energies of the short-range interactions are usually modelled as a combination of the attractive London dispersion interaction and the repulsive exchange forces that arise from the Pauli exclusions principle [52]. These are often forms such as shown below for the Lennard-Jones potential model [53],

$$E_{\text{non-boned}}(r) = E_{\text{repulsive}} + E_{\text{attractive}} = \frac{A}{r^{12}} - \frac{B}{r^6} = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \tag{2.53}$$

where, r is the distance between two particles, A and B are interaction dependent parameters, and σ and ϵ are simple reformations of these parameters,

$$A = 4\varepsilon\sigma^{12} \quad B = 4\varepsilon\sigma^{6}. \tag{2.54}$$

Figure 2.20 shows each component of the Lennard-Jones potential model for atoms of argon, using parameters for A and B determined by Rahman [54]. The Lennard-Jones is not the

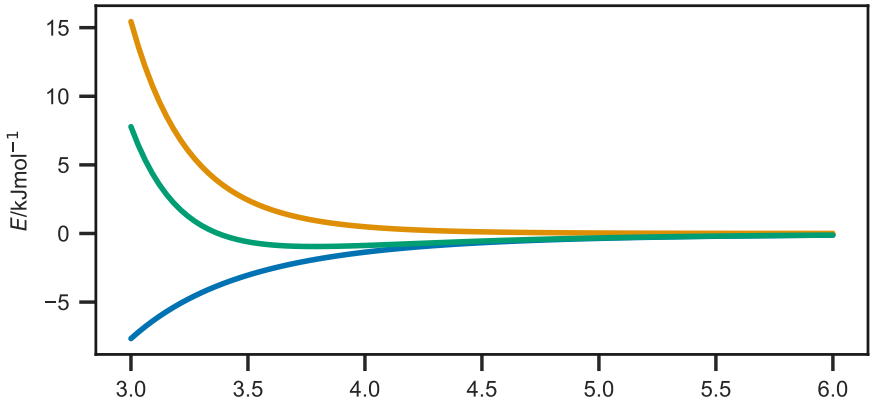


FIGURE 2.20: The form of each component; attractive (blue), repulsive (orange), of the Lennard-Jones potential model (green) for argon, using parameters from Rahman [54].

only potential model that may be used for the modelling of the short-range non-bonded interactions, others such as the Buckingham and Morse potentials exist [55, 56]. However, the Lennard-Jones model has been used heavily in this work.

While the short-range interactions are accounted for by a function such as the Lennard-Jones potential model, the potential energy of the long-range electrostatic interactions are usually modelled, more consistently, using Coulomb's law for classical electrostatic interaction between point particles [57, 58],

$$E_{\text{Coulomb}}(r) = \frac{1}{4\pi\varepsilon_0}\frac{q_i q_j e^2}{r^2}, \tag{2.55}$$

where, r is the distance between the two particles, ε_0 is the dielectric permittivity of the vacuum, e is the charge of the electron, and q_i and q_j are the electronic charges on each of the particles. It is clear that when q_i and q_j have the opposite signs Coulomb's law is always attractive.

An example of a very large classical simulation would be $\sim$ 3 million atoms [59]. However, this is still only 1.8×10^{-16} mol which is not remotely realistic as a simulation of a *real* system. A common method to allow for the apparent simulation of a much larger system is the use of periodic boundary conditions. This is where a boundary condition is applied to the edges of the simulation cell, such as to mimic an infinite system, assuming that the simulation cell is surrounded by identical images of itself (Figure 2.21). Using the periodic boundary condition means that atomic diffusion is conserved as when an atom reaches the edge of the simulation cell, it will appear on the other side such that it came from the adjacent periodic time. The use of a periodic boundary condition is particularly powerful in the simulation of homogenous systems, such as liquids. However, the periodicity may result in unexpected results for particular systems, as is discussed in Chapter 4.

The cut-off is another important factor for classical simulation, this is the distance after which the energy between two particles is considered to be zeros. Therfore for distances greater than the cut-off, it is not necessary to calculate the energy between the two particles as it is taken to be zero, increasing the computational efficiency. Code Block 2.2 gives an example of some code that could be used to calculate the Lennard-Jones energy of an atomistic system, where both the periodic boundary condition and the energy cut-off distance are considered.

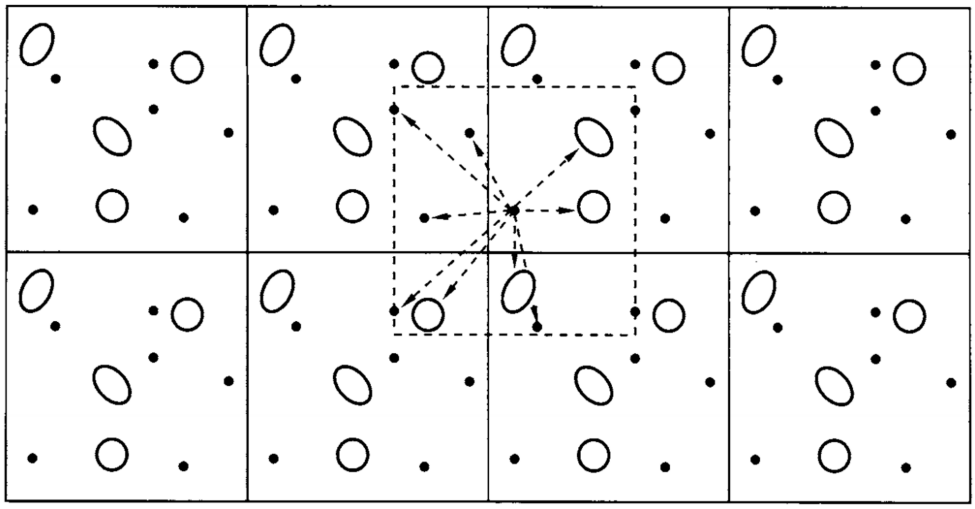


FIGURE 2.21: A graphical representation of the periodic boundary conditions. Reprinted by permission from Elsevier© from Reference [60].

CODE BLOCK 2.2: Code that may be used to generate the Lennard-Jones energy for a given atomistic system, which accounts for the periodic boundary condition and the energy cut-off distance.

```
import numpy as np


def lj_energy(coordinates, cell, cut_off, A, B):
    """
    Calculates the potential energy arising from the Lennard-Jones
    potential model from a series of atomistic particles.

    Parameters
    ----------
    coordinates: float, array-like (N, 3)
        An array of the x, y, and z coordinates for each of the N
        particles in the simulation.
    cell: float, array-like
        An array of length 3 containing the simulation cell vectors.
    cut_off: float
        The potential energy cut_off value for the simulation.
    A: float
        The A parameter in the Lennard-Jones potential model.
    B: float
        The B parameter in the Lennard-Jones potential model.

    Returns
    -------
    float, array-like
        An array of length N containing the energy of each particle
        in the simulation.
    """
    energy = np.zeros((coordinates.shape[0]))
    for i in range(coordinates.shape[0] - 1):
        for j in range(i + 1, coordinates.shape[0]):
            d = coordinates[j] - coordinates[i]
            d = d % cell
            r = np.sqrt(np.sum(np.square(d)))
            if r > cut_off:
                continue
            else:
                energy[i] += A / np.power(r, 12) - B / np.power(r, 6)
                energy[j] += A / np.power(r, 12) - B / np.power(r, 6)
    return energy
```

The use of the periodic boundary condition may be problematic for systems containing long-range interactions, such as classical electrostatics, due to the fact that the range of the electrostatic interaction may be much greater than the size of half of the simulation cell, which is usually taken to be the energy cut-off distance. In order to avoid truncation artefacts, the Ewald summation is often used for the calculation of the electrostatic contribution to the potential energy [61]. The Ewald summation involves performing the summation of the contributing interaction energies in reciprocal space rather than in real space as is the case for the short-range interactions. Most modern molecular dynamics simulation software packages implement the Ewald summation using a particle mesh Ewald (PME) method [62].

2.3.2 Parameterisation

Section 2.3.1 introduced the idea of potential models that may be used to evaluate the potential energy of a given system, much quicker than methods that rely on the use of quantum mechanics. However, for these methods to be effective, it is important that the potential models used are able to model the system under study accurately. This is achieved initially by selecting the correct potential model for a given interaction, and then by ensuring that the interaction-dependent parameters are accurate for a given interaction. The method of obtaining such parameters is referred to as *parameterising* the model. Model parameterisation is important for all types of potential models, for example it is necessary to determine the equilibrium bond length b_0 and the force constant K_b for a given covalent bonds, or the partial electrostatic charge that is present on a carbonyl oxygen atom when it interacts with the hydrogen atom from a neighbouring hydroxyl group.

Parameterisation of a potential model is usually achieved by comparison of fitting the potential model functions to energetic data obtained using a higher accuracy technique, such as quantum mechanical calculations or experimental methods. We will not dwell on the details of potential model parameterisation, however, it is important to note that the parameters used in molecular dynamics simulation are not absolute and depended heavily on the merits of the parameterisation method.

In the work, we focused heavily on the use of off-the-shelf potential models. The was done to ensure the easy reproducibility of the work. Off-the-shelf potential models are those that are determined to be applied to a wide range of chemical systems. An example includes the OPLS potential mode which was parameterised by comparison to quantum mechanical measurements and crystallographic data [63]. While these off-the-shelf potential models are useful for their ease-of-use, it is noted that often these forcefields may require optimisation for the particular system.

2.3.3 Coarse-graining

The atomistic simulation of very large systems, such as multiple surfactant micelles or large lipid monolayers, require a huge number of atoms. While computational efficiency improvements such as the periodic boundary condition or the energy cut-off distance are able to improve the time taken to simulate these systems, it is still not possible to produce physically meaningful simulations without including some other efficiency improvements.

This has led to the use of coarse-graining of molecules in simulations. This is the definition of super-atoms, in the place of groups of atoms, known as 'beading', some examples are shown for the MARTINI force field in Figure 2.22. Each of the super-atoms must correspond to the

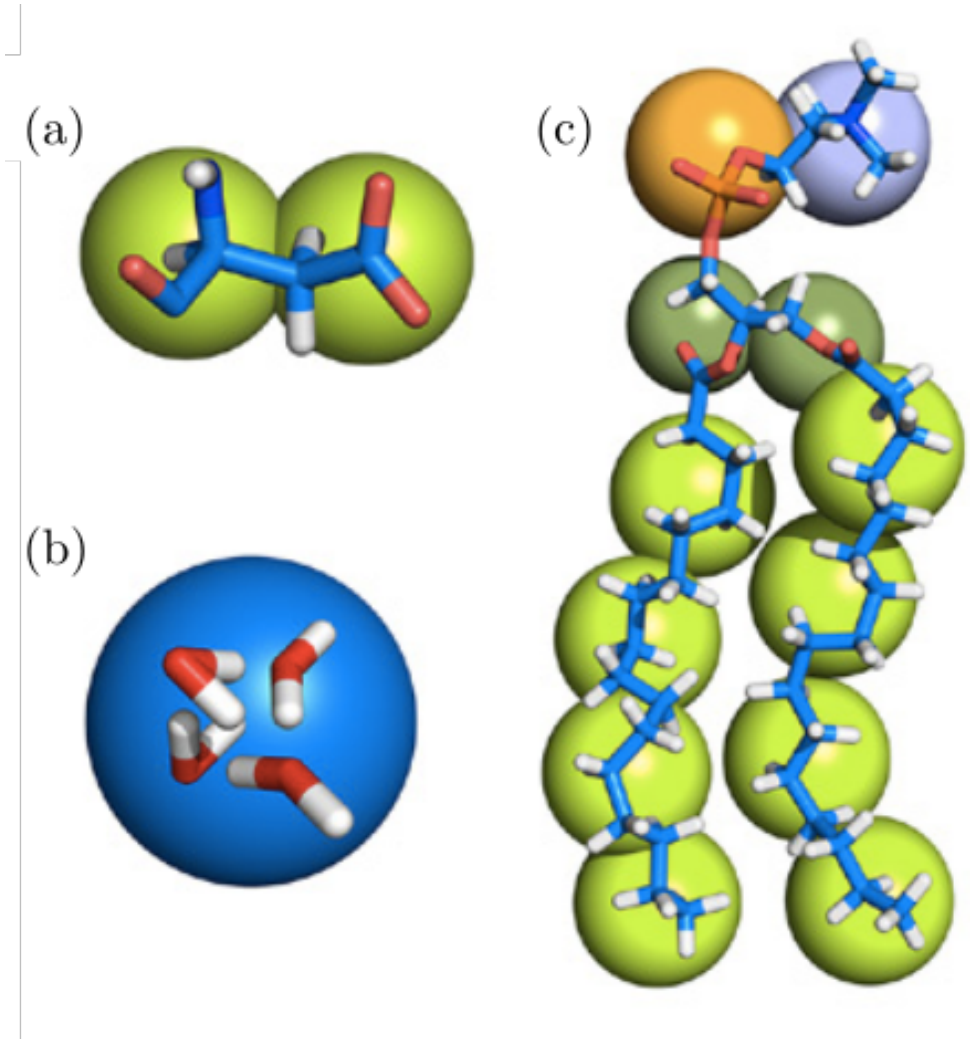


FIGURE 2.22: Three examples of the MARTINI coarse-graining mechanism for (a) aspertic acid, (b) a water cluster, and (c) a molecule of DPPC, from Reference [64].

chemistry of the underlying atoms. For example in the MARTINI potential model, there are five different apolar, carbon-like super-atoms that may be used dependent on the polarity of the carbon atoms that make up the super-atom. Additionally, there are thirteen other super-atom types that can be use to model polar, nonpolar, and charged atomic groups.

In addition to the computational benefit of having fewer particles in the simulation, and therefore requiring fewer integrations of the equations of motion, there is also the opportunity to increase the timestep length for the simulation [64]. This can be achieved as the highest frequency vibrations that must be modelled in the system are integrated out. For example, using a united atom potential model, where the hydrogen atoms have been integrated out, the time step may be larger than for the same all-atom system as it is no longer necessary to model the high-frequency C – H bond.

The technique of coarse-graining a molecule is a continuum, it can range from the integration of the hydrogen atoms into the heavier atoms to which they are bound, all the way to the treatment of entire molecules as a single 'bead', with the inclusion of an implicit solvent. The parameterisation of a coarse-grained potential model is carried out in much the same way as discussed in Section 2.3.2 for all-atom potential models. The coarse-grained parameters are determined by comparison with a higher-resolution technique, often this is all-atom

molecular dynamics simulations.

2.4 Optimisation & sampling methods

In this work, various computational methods have been applied to a series of important scattering problems. The aim of many modelling problems is to optimise a series of parameters such that a minimum is found in some parameter-dependent metric. While, in other circumstances, the aim is to sample the parametric search-space of a particular problem. The problem of parameter optimisation and sampling is a massive area of mathematics and computer science and is it not possible to introduce the whole field. Therefore, we will introduce three optimisation methods and two sampling methods that are applied within this work.

2.4.1 Single candidate optimisation methods

Single candidate methods are optimisation procedures that operate in a linear fashion, where there is a single sample that must be optimised. These type of methods are often more straightforward than the population methods discussed later, however frequently they are only suitable for the determination of local minima.

Gradient descent

The gradient descent is a simple, analytic optimisation algorithm that is capable of determining the local minimum of a given function. This method involves determining the local gradient of the function, at a given position and using this to define how the position should be changed. For some arbitrary, multi-dimensional function, $F(\mathbf{p})$, where $\mathbf{p}$ is the position, this can be described as,

$$\mathbf{a} \leftarrow \mathbf{a} - \alpha \frac{\partial F(\mathbf{p})}{\partial \mathbf{a}}, \tag{2.56}$$

where α is a constant that defines the step-size and $^{\partial F(\mathbf{p})}/_{\partial \mathbf{p}}$ is the first derivative of the multi-dimensional function. It is necessary to determine a suitable value for α for a given function, therefore often the gradient descent method is replaced with the, more computationally expensive, Newton-Raphson methods. The gradient descent method is shown applied to a Rosenbrock function [65], with a series of different step sizes in Figure 2.23 The gradient descent method is relatively simple to introduce, shown in Code Block 2.3, however, it is only capable of finding local minima and therefore is often unsuitable for situations where the global minima are desired.

2.4.2 Population optimisation methods

Population-based algorithms make use of a population of candidate solutions, compared to the single candidate shown in the gradient descent method above. This population of candidate solutions often have knowledge of the state of each other through some interaction method. The method of interaction is often used to characterise the algorithms, into evolutionary algorithms (such as differential evolution), and swarm intelligence algorithms (such as the particle swarm method detailed below) [66]. These population methods are usually

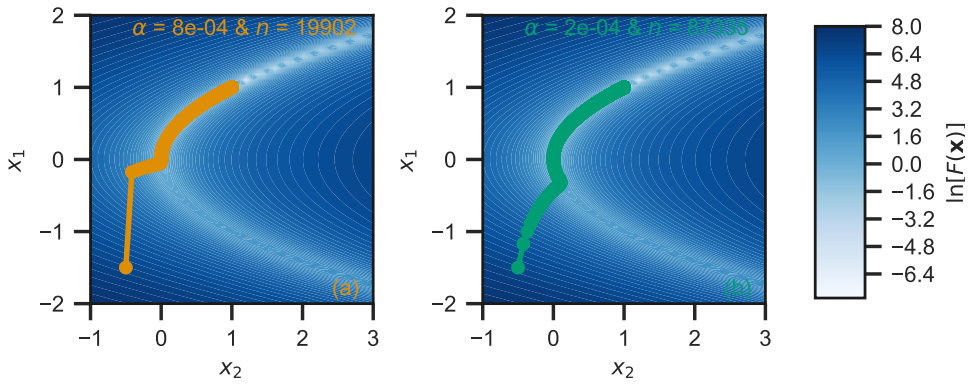


FIGURE 2.23: Example of the gradient descent method applied to a Rosenbrock function [65], where $a = 1$ and $b = 100$ (therefore the minima is at $(1, 1)$), n indicated the number of iterations required to minimise the values. Two different values (2×10^{-4} and 8×10^{-4}) for α are shown to emphasize the efficiency improvement that is possible with the optimisation.

CODE BLOCK 2.3: An example of a simple implementation for the Gradient descent, as applied to a two-dimensional function.

```
import numpy as np


def gradient_descent(init_x1, init_x2, dfx1, dfx2, alpha, threshold):
    """
    Applied the gradient descent method to a given two-dimensional
    function, the derivatives of which are dfx and dfy.

    Parameters
    ----------
    init_x1: float
        Initial value for x1.
    init_x2: float
        Initial value for x2.
    dfx1: function
        Derivative of function with respect to x1.
    dfx2: function
        Derivative of function with respect to x2.
    alpha: float
        Step-size for gradient descent steps.
    threshold: float
        The gradient at which fitting stops.

    Returns
    -------
    float, array_like
        The history of the parameters being fit. An array of
        shape m, n, where n is the number of iterations and
        m is the number of parameters.
    """
    history = np.array([])
    i = 1
    x1 = init_x1
    x2 = init_x2
    while (
        np.abs(dfx1(x1, x2)) > threshold
        and np.abs(dfx2(x1, x2)) > threshold
    ):
        history = np.append(history, np.array([x1, x2]))
        history = np.reshape(history, (i, 2))
        x1 = x1 - alpha * dfx1(x1, x2)
        x2 = x2 - alpha * dfx2(x1, x2)
        i += 1
    return history
```

more efficient at finding the global minimum for a given search space, than the single candidate methods described above.

Differential evolution

Differential evoulation (DE) is a common, iterative optimisation algorithm, that was first applied to the analysis of reflectometry and diffraction data by Wormington *et al.* [67]. Since then, it has proven very popular for the optimisation of reflectometry data with including in many common analysis programs [13, 14, 68–71]. The DE algorithm is designed to more ably determine the global minimum of a particular function [72].

DE is an example of a genetic algorithm, one that is designed to mimic the evolution processes observed in biology [73]. The method consists of two vectors, the parent population, $\mathbf{p}$, the offspring population, $\mathbf{o}$. These vectors are of a dimension $(i \times j)$, where i is the number of variables being optimised and j is the number of candidate solutions being used. The offspring population vector is created through some trial methods, many of these exist however we will discuss a simple classical trial method, details of other methods may be found in the work of Björck [68].

A classical trial method consists of two stages, mutation and recombination. The mutation stage involves performing some mutation on the parent population to create a mutant vector, $\mathbf{m}$, analogous to the mutation in biologically evolutionary theory. The magnitude of the mutation is dependent on the mutation constant, k_m,

$$\mathbf{m}_{i,j} = b_i + k_m(\mathbf{p}_{i,R1} - \mathbf{p}_{i,R2}), \tag{2.57}$$

where b_i is the best candidate solution in the parent population, and $\mathbf{p}_{i,R1}$ and $\mathbf{p}_{i,R2}$ are randomly choosen members of the parent population. The mutation constant can be considered as a control variable for the size of the search radius, with a large k_m corresponding to a larger search radius.

The recombination step creates the offspring population vector by taking a sample from either the parent population or mutant vectors with some frequency, which depends on the recombination constant, k_r,

$$\mathbf{o}_{i,j} = \begin{cases} \mathbf{m}_{i,j}, & \text{where } X < k_r \\ \mathbf{p}_{i,j}, & \text{otherwise} \end{cases} \tag{2.58}$$

where, $X \sim U[0, 1)$. The recombination constant controls the progress of the algorithm as it impacts the frequency with which mutation is introduced into the offspring population vector.

The final stage is to compare the offspring and parent population vectors, in the selection stage to create the new parent population for the next iteration. The selection stage comprises of using some figure of merit, ζ, to choose between the subunit from the offspring or parent population vector. In our example, that figure of merit may be the agreement between some experimental data and our model, or for the example in Figure 2.24 it is the value of the Ackley function [74], which we are trying to minimise.

$$\mathbf{p}_{*,j} \leftarrow \begin{cases} \mathbf{o}_{*,j}, & \text{where } \zeta_{\mathbf{o}_{*,j}} < \zeta_{\mathbf{p}_{*,j}} \\ \mathbf{p}_{*,j}, & \text{otherwise} \end{cases} \tag{2.59}$$

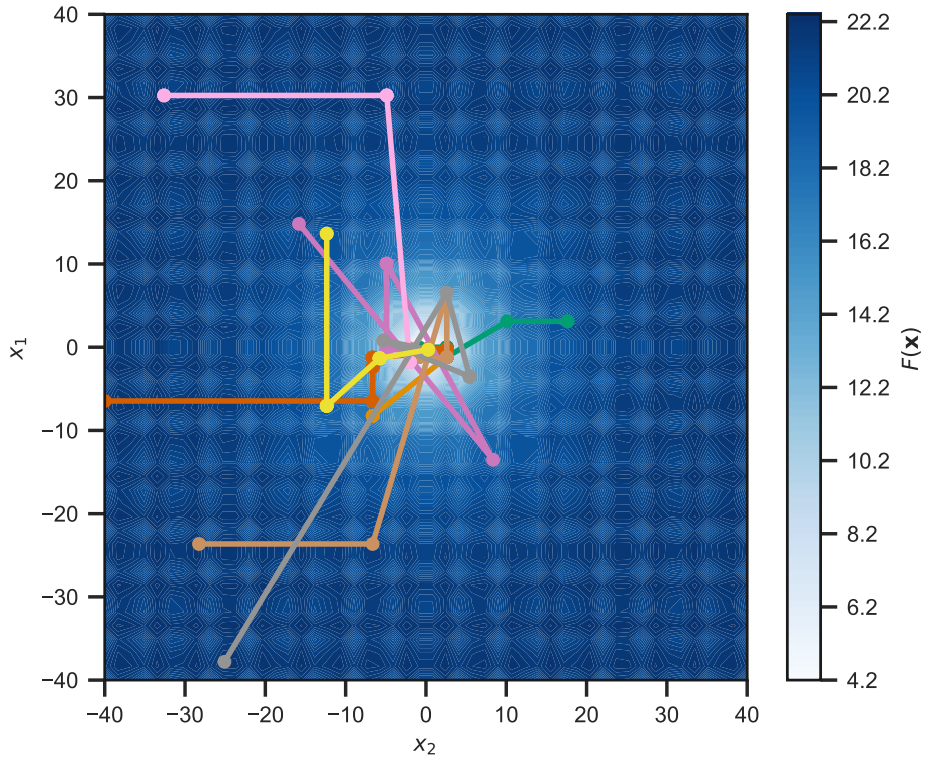


FIGURE 2.24: An example of a differential evolution (DE) algorithm as applied to an Ackley function [74], where $a = 20$, $b = 0.2$, and $c = 2\pi$ (a common function for assessing global optimisation methods). The mutation and recombination constant in this implementation are both 0.5. Each different coloured line represents a different candidate solution. The optimisation was stopped after 100 iterations had run.

CODE BLOCK 2.4: An example of a simple implementation for a DE algorithm as described by Björck [68].

```
import numpy as np
import mutation as mut
import recombination as recomb
import selection as sel


def differential_evolution(
    population, f, km, kr, bounds, stop, max_iter
):
    """
    An implementaiton of a differential evolution algorithm.

    Parameters
    ----------
    population: float, array_like
        The initial parent population vector. An array of
        shape (i, j), where i is the number of varibles and
        j is the population size.
    f: function
        The figure of merit function to be minimised.
    km: float
        The mutation constant.
    kr: float
        The recombination constant.
    bounds: float, array_like
        The minimum and maximum values for the bounds.
    stop: float
        The value at which the optimisation should stop.
    max_iter: int
        The maximum number of iterations that should be performed
        regardless of if stop is reached.

    Returns
    -------
    float, array_like
        The history of the parameters being fit. An array of
        shape (m, n, j), where n is the number of iterations,
        m is the number of parameters, and j is the population
        size.
    """
    history = np.array([population])
    best = population[:, np.argmin(f(population))]
    i = 0
    while f(best) > stop and i < max_iter:
        mutant = mut.mutation(population, best, km)
        offspring = recomb.recombination(population, mutant, kr)
        offspring[
            np.where(offspring >= bounds[1])
            or np.where(offspring < bounds[0])
        ] = np.random.uniform(bounds[0], bounds[1], 1)
        selected = sel.selection(population, offspring, f)
        history = np.append(history, selected)
        history = np.reshape(
            history, (i + 2, population.shape[0], population.shape[1])
        )
        population = np.array(selected)
        best = population[:, np.argmin(f(population))]
        i += 1
    return history
```

where, the $*$ notation indicates all objects in the given population, and $\zeta_{\mathbf{o}_{*,j}}$ and $\zeta_{\mathbf{p}_{*,j}}$ are the figures of merit for the offspring and population population candidate solutions respectively.

It is noted that it is often the case, in particular when optimising experimental data, that there should be some bounds applied to the variables within the populations. However, the DE algorithm may disregard these bounds due to the nature of the mutation step. Therefore, it is common in DE algorithms, where bounds must be set, that if the search space moves outside that expected it is necessary to reinitialise the parameter. An implementation of the DE algorithm is given programmatically in Code Block 2.4, where this reinitialisation is achieved by obtaining a new random number within the given bounds.

CODE BLOCK 2.5: The mutation step used in a classical trial method for a differential evolution algorithm, as described by Björck [68].

```
import numpy as np


def mutation(p, b, km):
    """
    Performs the mutation step, where the parent population vector
    is transforms to produce a mutant vector.

    Parameters
    ----------
    p: float, array_like
        The parent population vector. An array of shape (i, j),
        where i is the number of varibles and j is the population
        size.
    b: float, array_like
        The best member of the parent population. An array of
        size i, where i is the number of variables.
    km: float
        The mutation constant.

    Returns
    -------
    float, array_like
        The mutant vector. An array of shape (i, j), where i is
        the number of varibles and j is the population size.
    """
    m = np.zeros_like(p)
    R = np.random.randint(p.shape[1], size=(2, p.shape[1]))
    for j in range(p.shape[1]):
        m[:, j] = b + km * (p[:, R[0, j]] - p[:, R[1, j]])
    return m
```

CODE BLOCK 2.6: The recombination step used in a classical trial method for a differential evolution algorithm, as described by Björck [68].

```
import numpy as np


def recombination(p, m, kr):
    """
    The recombination step, where the mutant and parent population
    vectors are combined.

    Parameters
    ----------
    p: float, array_like
        The parent population vector. An array of shape (i, j),
        where i is the number of varibles and j is the population
        size.
    m: float, array_like
        The mutant vector. An array of shape (i, j), where i
        is the number of varibles and j is the population size.
    kr: float
        The recombination constant.

    Returns
    -------
    float, array_like
        The offspring population vector. An array of shape (i, j),
        where i is the number of varibles and j is the population
        size.
    """
    o = np.array(p)
    rand = np.random.rand(p.shape[0], p.shape[1])
    o[rand < kr] = m[rand < kr]
    return o
```

CODE BLOCK 2.7: The mutation step used in a differential evolution algorithm, as described by Björck [68].

```
import numpy as np


def selection(p, o, f):
    """
    The selection function, where the population member that
    return the minimum value for the figure of merit is brought
    forward to the next generation.

    Parameters
    ----------
    p: float, array_like
        The parent population vector. An array of shape (i, j),
        where i is the number of varibles and j is the population
        size.
    o: float, array_like
        The offspring population vector. An array of shape (i, j),
        where i is the number of varibles and j is the population
        size.
    f: function
        The figure of merit function to be minimised.

    Returns
    -------
    float, array_like
        The new parent population vector. An array of shape (i, j),
        where i is the number of varibles and j is the population
        size.
    """
    new_p = np.array(p)
    for j in range(p.shape[1]):
        p_fom = f(p[:, j])
        o_fom = f(o[:, j])
        if o_fom < p_fom:
            new_p[:, j] = o[:, j]
    return new_p
```

Particle swarm

Particle swarm optimisation is a type of swarm intelligence population-based optimisation method. This optimisation method was originally developed by Kennedy, Eberhart, and Shi for the simulation of social organisms such as bird flocks [75, 76]. Particle swarm methods are particularly suitable for the optimisation, and sampling, of parametric search-spaces with a large number of similar minima, and therefore is useful for the study of the self-assembly of soft matter materials (Section 5).

These methods consist of a population vector, similar to that described for the differential evolution, that moves around the parametric search-space. The motions of these 'particles' are influenced by the positions of the other particles in the vector [77]. It is anticipated that this will lead the swarm to optimise the function under investigation.

Particles in the swarm are under the influence of two elastic forces. The first attracts the particle to the best location in the search-space that that particle has found, while the other attracts the particle to the best search-space location found by any particle of the swarm. The magnitude of these forces is randomised but modulated by a pair of acceleration coefficients, ψ_p that influences the attraction towards the personal best location, and ψ_g that influences the attraction to the global best location. The position of a particle changes between iterations of the algorithm based on the following relation,

$$\mathbf{p}_{*,j} \leftarrow \mathbf{p}_{*,j} + \mathbf{v}_{*,j}, \tag{2.60}$$

where, $\mathbf{p}_{*,j}$ is the position of the particle, and $\mathbf{v}_{*,j}$ is the velocity of the particle. This velocity is determined as shown below,

$$\mathbf{v}_{*,j} \leftarrow \omega\mathbf{v}_{*,j} + \psi_g R1(\mathbf{g}_* - \mathbf{p}_{*,j}) + \psi_p R2(\mathbf{s}_{*,j} - \mathbf{p}_{*,j}), \tag{2.61}$$

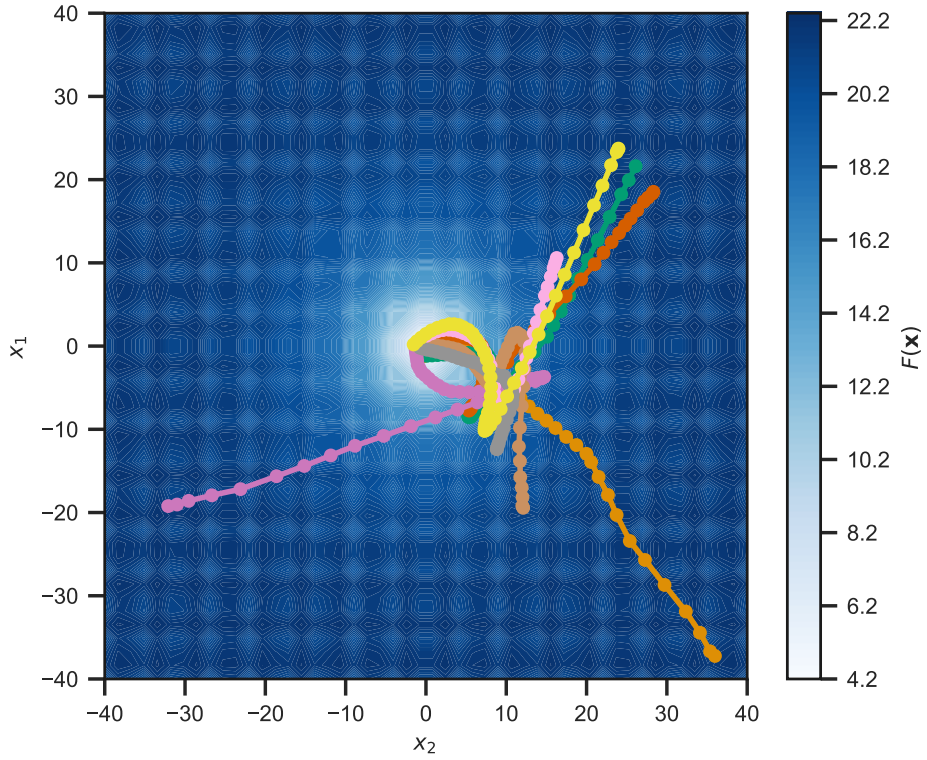


FIGURE 2.25: An example of a particle swarm optimisation as applied to an Ackley function [74], where $a = 20$, $b = 0.2$, and $c = 2\pi$. For the particle swarm, the following parameters were used $\omega = 0.9$, $\psi_g = 0.05$, and $\psi_p = 0.05$. Each different coloured line represents a different candidate solution. The optimisation was stopped after 100 iterations had run.

where, ω a constant known as the interia weight, $R1 \sim U[0, 1)$ and $R2 \sim U[0, 1)$ are random numbers, $\mathbf{g}_*$ is the best position occupied by any particle in the swarm and $\mathbf{s}_{*,j}$ is the person best for the particle j.

Figure 2.25 shows an example of the particle swarm optimisation in action, applied to the Ackley function [74]. Code Block 2.8 shows a functional programmatic implementation of a particle swarm optimisation algorithm.

2.4.3 Markov chain Monte-Carlo

Markov chain Monte-Carlo (MCMC) is a sampling methodology, derived from direct sampling Monte-Carlo [78]. The aim of an MCMC algorithm is to sample a probability distribution, enabling Bayesian inference, when parameters are described in terms of their degree of probability [79]. Similar to molecular dynamics, in practical terms, MCMC should not be used on a system that is not already optimised. Generally, the approach would be to optimise using, for example, one of the approaches described above, then to use MCMC or molecular dynamics to sample the appropriate search-space. For example, in this work MCMC is used extensively, following the optimisation of a reflectometry model using a differential evolution algorithm, to quantify the inverse uncertainties of the model. In this context, the inverse uncertainty is the uncertainty on a given parameter within the model. In addition to being able to give information about the inverse uncertainties, MCMC also

CODE BLOCK 2.8: An example of the particle swarm optimisation algorithm [77].

```
import numpy as np


def particle_swarm(position, f, omega, psig, psip, stop, max_iter):
    """
    An implementaiton of a particle swarm optimisation algorithm.

    Parameters
    ----------
    position: float, array_like
        The initial position vector. An array of shape (i, j),
        where i is the number of varibles and j is the
        population size.
    f: function
        The figure of merit function to be minimised.
    omega: float
        The interia weight.
    psig: float
        The global acceleration coefficient.
    psip: float
        The personal acceleration coefficient.
    stop: float
        The value at which the optimisation should stop.
    max_iter: int
        The maximum number of iterations that should be performed
        regardless of if stop is reached.

    Returns
    -------
    float, array_like
        The history of the parameters being fit. An array of
        shape (m, n, j), where n is the number of iterations,
        m is the number of parameters, and j is the population
        size.
    """
    history = np.array([position])
    velocity = np.zeros_like(position)
    g_best = position[:, np.argmin(f(position))]
    p_best = np.array(position)
    i = 0
    while f(g_best) > stop and i < max_iter:
        for j in range(velocity.shape[1]):
            velocity[:, j] = (
                omega * velocity[:, j]
                + psig * np.random.rand() * (g_best - position[:, j])
                + psip
                * np.random.rand()
                * (p_best[:, j] - position[:, j])
            )
            position[:, j] = position[:, j] + velocity[:, j]
        history = np.append(history, position)
        history = np.reshape(
            history, (i + 2, position.shape[0], position.shape[1])
        )
        test_g_best = position[:, np.argmin(f(position))]
        if f(test_g_best) < f(g_best):
            g_best = test_g_best
        test_p_best = np.array(position)
        for j in range(position.shape[1]):
            if f(test_p_best[:, j]) < f(p_best[:, j]):
                p_best[:, j] = test_p_best[:, j]
        i += 1
    return history
```

offers a more profound understanding of the correlations present between the different parameters [80].

The aim of MCMC is to only sample configurations of a given function that are within the experimental uncertainty. Figure 2.26 shows an example of the possible output that may be obtained from the application of an MCMC sampling method. This was generated using a Metropolis-Hastings algorithm [81, 82], shown in Code Block 2.9. Initially, a Levenberg–Marquardt algorithm [83, 84] was used to optimise the positions and integral of the two Gaussian functions that make up the data. The MCMC was used to sample the values that were within the experimental uncertainty.

Once an optimised solution, θ, is obtained, the figure of metric is calculated, in Code Block 2.9 this is the agreement between the model and the experimental data, χ^2, where,

$$\chi^2 = \sum \frac{(y_{\text{exp}} - y_{\text{calc}})^2}{\mathrm{d}y_{\text{exp}}}, \tag{2.62}$$

and y_{exp} is the experimental data, and $\mathrm{d}y_{\text{exp}}$ the uncertainty in the data, while y_{calc} is the model solution. Some random pertubation is then applied to the optimised solution,

$$\Theta = \theta + aR, \tag{2.63}$$

where $R \sim N(0,1)$. A new χ^2 is found for Θ, and the probablity that this transition will occur is found,

$$p = \exp\left(\frac{-\chi^2(\Theta) + \chi^2(\theta)}{2}\right). \tag{2.64}$$

This probability is then compared with a random number $n \sim U[0,1)$, and if n is less than the probability, the new solution is stored,

$$\theta \leftarrow \Theta. \tag{2.65}$$

This process is repeated until some desired number of samples has been obtained. It should be noted that in the event on a poorly optimised initial value of θ, it may be necessary to 'burn' (that is to ignore) the first series of solutions while the MCMC algorithm settles into the search-space.

2.4.4 Molecular dynamics

Section 2.3 introduced classical potential models as a method for the determination of the energy of a given chemical system. Any of the optimisation methods discussed above could be used alongside these classical potential models to try and find the energetic minimum structure for the system or to sample the potential energy landscape. However, it is often the case that we are interested in the dynamically relevant structure at a given temperature for some system. This is where molecular dynamics simulations are a useful and important tool.

Forces and accelertions

The aim of a molecular dynamics simulation is to probe the positions, velocities, and accelerations on each of the atoms, or coarse-grained particles, as a simulation progresses. The

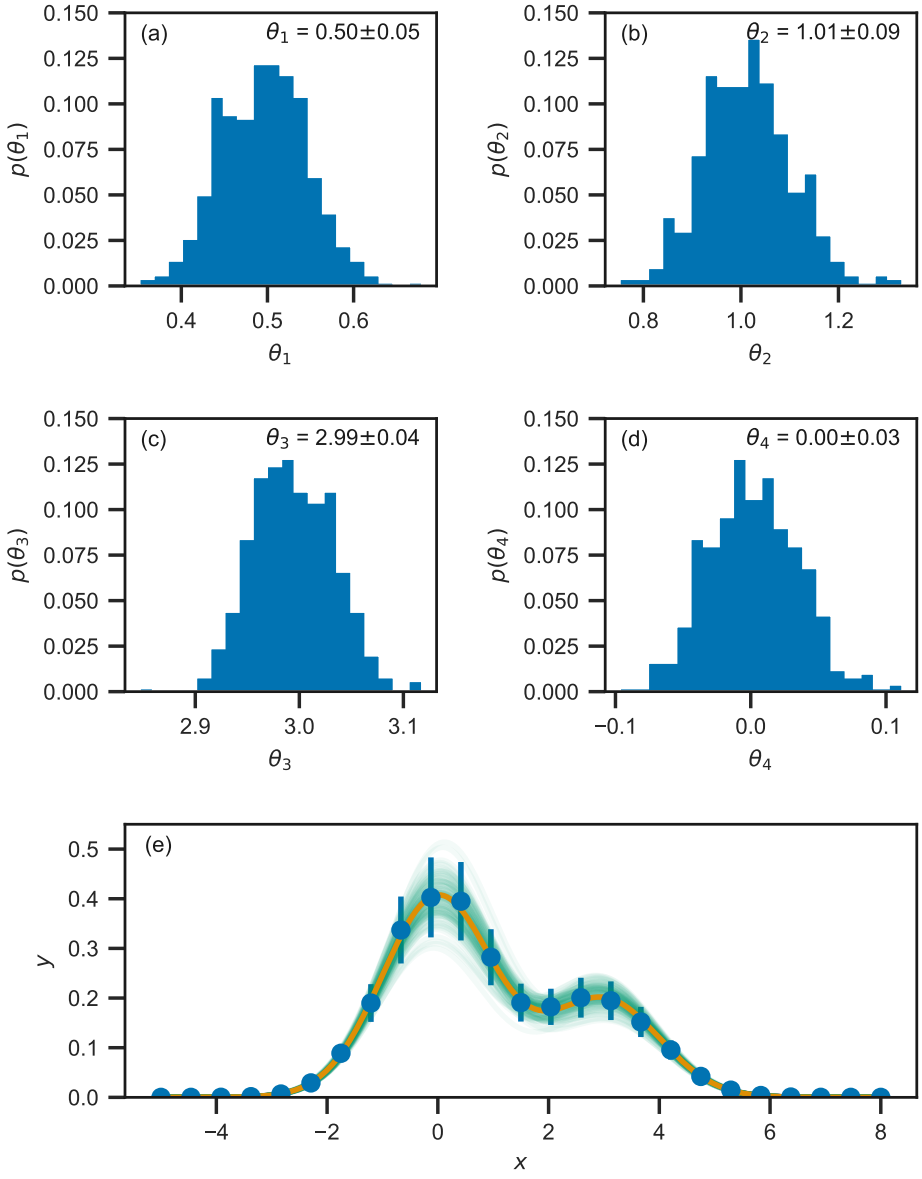


FIGURE 2.26: An example of a four variable (two nearby Gaussian functions of different sizes with added random noise and some fractional uncertainty) problem probed using a MCMC method, using values of $a = 0.1$, θ_1 and θ_2 correspond to the integral of the Gaussian function, while θ_3 and θ_4 indicate their positions; (a)-(d) histograms of the probability distribution function for each of the varibles, and (e) the data (blue circles), the optimised solution (orange line), and a series of probable solutions (green lines) showing the variability present in the data uncertainty.

CODE BLOCK 2.9: An example of the Metropolis-Hastings MCMC algorithm [81, 82].

```
import numpy as np


def mcmc(theta, f, a, data, iterations, nburn):
    """
    A simple implemation of the Metropolis-Hasting MCMC algorithm

    Parameters
    ----------
    theta: float, array_like
        Initial values for the variables. An array of length n,
        where n is the number of variables.
    f: function
        The function being fit to the data.
    a: float, array_like
        The step size. An array of length n, where n is the number
        of variables.
    data: float, array_like
        The data being assessed. An array of shape  (m, 3)
        containing x, y, and dy.
    iterations: int
        The number of accepted iterations to obtain.
    nburn: int
        The number of accepted iterations to ignore during the
        burn-in phase.

    Returns
    -------
    float, array_like
        The history of the parameters being fit. An array of
        shape (m, n), where n is the number of iterations,
        and m is the number of parameters.
    """
    accepted = np.array([])
    calc_y = f(data[0], theta)
    chi2 = np.sum(np.square((data[1] - calc_y) / data[2]))
    i = 0
    while i < iterations:
        new_theta = theta + a * np.random.randn(theta.size)
        new_calc_y = f(data[0], new_theta)
        new_chi2 = np.sum(np.square((data[1] - new_calc_y) / data[2]))
        prob = np.exp((-new_chi2 + chi2) / 2)
        n = np.random.rand()
        if n < prob:
            i += 1
            theta = new_theta
            chi2 = new_chi2
            if i > nburn:
                accepted = np.append(accepted, theta)
                accepted = np.reshape(
                    accepted, (i - nburn, theta.size)
                )
    return accepted
```

acceleration on a given particle, $\mathbf{a}$ is defined by the force on that particle, $\mathbf{f}$, in agreement with Newton's second law of motion,

$$\mathbf{f} = m\mathbf{a}, \tag{2.66}$$

where, m is the mass of the particle. In order to determine the acceleration on the particle, it is necessary to know the force on that particle. The force, f, is a function of the potential energy, E, as found from a classical potential, of that atom,

$$f(r) = \frac{-\partial E_{\text{total}}(r)}{\partial r}, \tag{2.67}$$

where, r is the configuration of the atoms. Which is to say that, the force is the negative of the first derivative of the energy with respect to the atomic configuration. The force found from Equation 2.67 is a scalar, however, we are interested in the force vector in Equation 2.66. To determine the force in a given direction, it is necessary to find the product of the force, f, and the unit vector in that direction,

$$\mathbf{f}_x = f\hat{\mathbf{r}}_x, \quad \text{where } \hat{\mathbf{r}}_x = \frac{r_x}{|\mathbf{r}|}, \tag{2.68}$$

where r_x is the atomic configuration in the x-dimension, and $|\mathbf{r}|$ is the magnitude of the atomic configuration vector.

Integration

The potential model, which we define for a given system, allows for the calculation of the acceleration on each particle in that system. The next step is to use this acceleration to iterate through the trajectory of our system. This is achieved by applying Newtonian equations of motion, for example in the Velocity-Verlet algorithm [85].

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \mathbf{v}(t)\Delta t + \frac{1}{2}\mathbf{a}(t)\Delta t^2, \tag{2.69}$$

$$\mathbf{v}(t + \Delta t) = \mathbf{v}(t) + \frac{1}{2}\big[\mathbf{a}(t) + \mathbf{a}(t + \Delta t)\big]\Delta t, \tag{2.70}$$

where, $\mathbf{x}$ is the position the particle $\mathbf{v}$ is the particle's velocity, and $\mathbf{a}$ is the particle's acceleration, while t is current simulation time and Δt is the timestep. These equations constitute the Velocity-Verlet algorithm,

1. calculate the force (and therefore the acceleration) on each particle (Equations 2.66 & 2.67),
2. find the position of the particle after some timestep (Equation 2.69),
3. determine the new velocity for each particle, based on the average acceleration at the current and new positions (Equation 2.70),
4. overwrite the old acceleration values with the new ones,
5. go to 1.

Following the initial relaxation of the particles to some equilibrium, this algorithm may be iterated as many times as is required to obtain sufficient statistics for the measurement quantity of interest, e.g. particle positions for structural techniques such as elastic scattering.

The above analytical process is known as the integration step, and the Velocity-Verlet is the integrator. If the size of the timestep Δt is too large, the acceleration that is calculated for the step will not be accurate, as the forces on the atoms will change too significantly during it. Therefore, the values of the timestep is usually on the order of $10 \times 10^{-15}\,\mathrm{s}$ (fs). This means that in order to simulate a single nanosecond of "real-time" molecular dynamics, the integrator must be solved one million times. This can be slow for very large systems, leading to an interest in coarse-grained simulations that result in fewer particles to determine the forces for (speeding up the integration step), but also enable to use of larger timesteps (so fewer integrations must be solved) [86, 87].

Initialisation

The above discussion ignored two aspects that are necessary to run a molecular dynamics simulation, both of which as associated with the original configuration of the system; the original particle positions and velocities.

The particle positions are usually taken from some library, for example for the simulation of a protein, often the protein data bank [88] is a useful resource. Small molecules may be configured by hand using graphical programs such as Jmol [89]. These small molecules may be built into complex, multicomponent structures using software such as the Packmol package [90]. The importance of this initial structure cannot be overstated, for example, if the initial structure in a molecular dynamics simulation is unrepresentative of the equilibrium structure, it may take a large amount of simulation time before the equilibrium structure is obtained, possibly much longer than could be reasonably simulated.

The initial particle velocities are obtained in a much more general fashion. They are selected randomly, and then scaled such that the kinetic energy, E_K, of the system agrees with a defined temperature, T,

$$E_K = \sum_{i=1}^{N} \frac{m_i |\mathbf{v}_i|^2}{2} = \frac{3}{2} N k_B T, \tag{2.71}$$

where, m_i and $\mathbf{v}_i$ are the masses and velocities of the particles, N is the number of particles, and k_B is the Boltzmann constant.

Ensembles

The above algorithm details a simulation that makes use of an NVE ensemble, a simulation where the number of particles (N), the volume of the system (V), the energy of the system (E) are all kept constant. However, this is not the only simulation ensemble that is available, within this work two other ensembles have been used extensively,

- the NVT (canonical) ensemble; this is similar to the NVE ensemble except the simulation temperature is controlled via a thermostat,
- the NPT (isothermal-isobaric); this ensemble is similar to the NVT ensemble, however, the system volume is allowed to vary while the overall system pressure is held constant using a barostat.

Thermostating involves controlling the kinetic energy of the particles (Equation 2.71) such that the simulation temperature is kept at a predefined value. There are a variety of methods for thermostating a molecular dynamics simulation, such as the Andersen, Nosé-Hoover, or Berendsen methods [91–94]. However, the most straightforward to describe, and that

implemented in the `pylj` software (discussed in detail in Chapter 6) [95, 96] is a velocity rescaling [97]. This is where the velocities for a random subset of the particles, $\mathbf{v}_i$ are adapted based on the following relation,

$$\mathbf{v}_i \leftarrow \mathbf{v}_i \sqrt{\frac{T_{\text{target}}}{\bar{T}}} \tag{2.72}$$

where, T_{target} is the target temperature, and $\bar{T}$ is the average simulation temperature.

The use of a barostat to control the simulation pressure usually involves varying the simulation cell parameters and the distances between the particles. This would in a similar way to thermostating, where the simulation dimensions are scaled by a value in an effort to control the pressure. The barostating methods are similar to the thermostating methods with Andersen, Nosé-Hoover, and Berendsen methods. However, there is also the Parrinello-Rahman barostat which allows for independent control of the different cell dimensions giving control of stress in addition to pressure [98].

These optimisation and sampling methods were used in a variety of different applications within this work, firstly differential evoluation optimisation and MCMC sampling are used in Chapter 3 in the study of a chemically-consistent modelling approach to X-ray and neutron reflectometry analysis. Molecular dynamics simulation are investigated as a possible tool to assist in the analysis of reflectometry and grazing incidence small angle scattering. Finally, the particle swarm optimisation is applied for the efficient determination of a micelle structure for ftting small angle scattering data.

2.5 References

[1] H. Schnablegger and Y. Singh, *The SAXS Guide: Getting Acquainted with the Principles*, Anton Paar GmbH, Graz, AT, 4th edn., 2017.

[2] *MetalJet X-Ray Source Technology*, `https://www.excillum.com/technology/`, (accessed 2018-12-06).

[3] M. C. Garcia-Gutierrez and D. R. Rueda, in *Applications of Synchrotron Light to Scattering and Diffraction in Materials and Life Sciences*, ed. M. Gomez, A. Nogales, M. C. Garcia-Gutierrez and T. A. Ezquerra, Springer-Verlag Berlin Heidelberg, Berlin, DE, 2009, pp. 1–22.

[4] D. S. Sivia, *Elementary Scattering Theory: For X-Ray and Neutron Users*, Oxford University Press, Oxford, UK, 2011.

[5] *ILL: Neutron for Science: Technical Characteristics*, `https://www.ill.eu/reactor-environment-safety/high-flux-reactor/technical-characteristics/`, (accessed 2016-08-08).

[6] *ISIS – How ISIS Works*, `https://www.isis.stfc.ac.uk/Pages/What-does-ISIS-Neutron-Muon-Source-do.aspx`, (accessed 2018-09-25).

[7] V. Garcia Sakai and A. Arbe, *Curr. Opin. Colloid Interface Sci.*, 2009, **14**, 381–390.

[8] B. Farago, *Curr. Opin. Colloid Interface Sci.*, 2009, **14**, 391–395.

[9] A. R. McCluskey and K. J. Edler, *Curr. Org. Chem.*, 2018, **22**, 750–757.

[10] F. Abelès, *Ann. Phys.*, 1948, **12**, 504–520.

[11] L. G. Parratt, *Phys. Rev.*, 1954, **95**, 359–369.

[12] F. Foglia, M. J. Lawrence and D. J. Barlow, *Curr. Opin. Colloid Interface Sci.*, 2015, **20**, 235–243.

[13] A. R. J. Nelson and S. W. Prescott, *J. Appl. Crystallogr.*, 2019, **52**, 193–200.

[14] A. R. J. Nelson, *J. Appl. Crystallogr.*, 2006, **39**, 273–276.

[15] A. V. Hughes, *RasCAL*, `https://sourceforge.net/projects/rscl/`, (accessed 2016-08-08).
[16] Y. Gerelli, *J. Appl. Crystallogr.*, 2016, **49**, 330–339.
[17] Y. Gerelli, *J. Appl. Crystallogr.*, 2016, **49**, 712–712.
[18] B. T. M. Willis and C. J. Carlile, *Experimental Neutron Scattering*, Oxford University Press, Oxford, UK, 2009.
[19] *ISIS – SANS2D*, `https://www.isis.stfc.ac.uk/Pages/Sans2d.aspx`, (accessed 2018-09-25).
[20] I. Grillo, in *Soft-Matter Characterization*, ed. R. Borsali and R. Pecora, Springer, New York, USA, 2008, pp. 723–782.
[21] K. J. Edler and D. T. Bowron, *Curr. Opin. Colloid Interface Sci.*, 2015, **20**, 227–234.
[22] S. Skou, R. E. Gillilan and N. Ando, *Nat. Protoc.*, 2014, **9**, 1727–1739.
[23] J. S. Pedersen, in *Neutron, X-Rays and Light. Scattering Methods Applied to Soft Condensed Matter*, ed. T. Zemb and P. Lindner, Elsevier Science B.V., Amsterdam, NL, 2002, pp. 381–390.
[24] *SASview for Small Angle Scattering*, `http://www.sasview.org`, (accessed 2016-10-26).
[25] *SASfit*, `https://kur.web.psi.ch/sans1/SANSSoft/sasfit.html`, (accessed 2018-11-11).
[26] R. Klein, in *Neutron, X-Rays and Light. Scattering Methods Applied to Soft Condensed Matter*, ed. T. Zemb and P. Lindner, Elsevier Science B.V., Amsterdam, NL, 2002, pp. 351–380.
[27] J. B. Hayter and J. Penfold, *Mol. Phys.*, 1981, **42**, 109–118.
[28] P. Debye, *Ann. Phys.*, 1915, **351**, 809–823.
[29] D. I. Svergun, *Acta Crystallogr. A*, 1994, **50**, 391–402.
[30] M. C. Watson and J. E. Curtis, *J. Appl. Crystallogr.*, 2013, **46**, 1171–1177.
[31] P. Müller-Buschbaum, S. V. Roth, M. Burghammer, E. Bauer, S. Pfister, C. David and C. Riekel, *Physica B*, 2005, **357**, 148–151.
[32] H. Zhang, W. Wang, S. Mallapragada, A. Travesset and D. Vaknin, *J. Phys. Chem. C*, 2017, **121**, 15424–15429.
[33] P. Müller-Buschbaum, in *Applications of Synchrotron Light to Scattering and Diffraction in Materials and Life Sciences*, ed. M. Gomez, A. Nogales, M. C. Garcia-Gutierrez and T. A. Ezquerra, Springer-Verlag Berlin Heidelberg, Berlin, DE, 2009, pp. 61–89.
[34] P. Müller-Buschbaum, J. S. Gutmann, R. Cubitt and W. Petry, *Physica B*, 2004, **350**, 207–210.
[35] C. Nicklin, T. Arnold, J. Rawle and A. Warne, *J. Synchotron Radiat.*, 2016, **23**, 1245–1253.
[36] A. Buffet, A. Rothkirch, R. Döhrmann, V. Körstgens, M. M. Abul Kashem, J. Perlich, G. Herzog, M. Schwartzkopf, R. Gehrke, P. Müller-Buschbaum and S. V. Roth, *J. Synchotron Radiat.*, 2012, **19**, 647–653.
[37] A. Hexemer, W. Bras, J. Glossinger, E. Schaible, E. Gann, R. Kirian, A. MacDowell, M. Church, B. Rude and H. Padmore, *J. Phys. Conf. Ser.*, 2010, **247**, 012007.
[38] A. Hexemer and P. Müller-Buschbaum, *IUCrJ*, 2015, **2**, 106–125.
[39] B. Lee, I. Park, H. Park, C.-T. Lo, T. Chang and R. E. Winans, *J. Appl. Crystallogr.*, 2007, **40**, 496–504.
[40] P. Busch, M. Rauscher, D.-M. Smilgies, D. Posselt and C. M. Papadakis, *J. Appl. Crystallogr.*, 2006, **39**, 433–442.
[41] P. Busch, D. Posselt, D.-M. Smilgies, M. Rauscher and C. M. Papadakis, *Macromolecules*, 2007, **40**, 630–640.
[42] A. Naudon, D. Babonneau, D. Thiaudière and S. Lequien, *Physica B*, 2000, **283**, 69–74.
[43] R. Lazzari, *J. Appl. Crystallogr.*, 2002, **35**, 406–421.
[44] J. Burle, C. Durniak, J. M. Fisher, M. Ganeva, G. Pospelov, W. van Herck and J. Wuttke,

BornAgain - Software for Simulating and Fitting X-Ray and Neutron Small-Angle Scattering at Grazing Incidence., `www.bornagainproject.com`, (accessed 2019-03-04).
[45] S. T. Chourou, A. Sarje, X. S. Li, E. R. Chan and A. Hexemer, *J. Appl. Crystallogr.*, 2013, **46**, 1781–1795.
[46] P. Schurtenberger, in *Neutron, X-Rays and Light. Scattering Methods Applied to Soft Condensed Matter*, ed. T. Zemb and P. Lindner, Elsevier Science B.V., Amsterdam, NL, 2002, pp. 145–170.
[47] M. T. Ivanović, L. K. Bruetzel, J. Lipfert and J. S. Hub, *Angew. Chemie Int. Ed.*, 2018, **57**, 5635–5639.
[48] E. Scoppola and E. Schneck, *Curr. Opin. Colloid Interface Sci.*, 2018, **37**, 88–100.
[49] A. P. Dabkowska, L. E. Collins, D. J. Barlow, R. Barker, S. E. McLain, M. J. Lawrence and C. D. Lorenz, *Langmuir*, 2014, **30**, 8803–8811.
[50] J. S. Hub, *Curr. Opin. Struct. Biol.*, 2018, **49**, 18–26.
[51] J. L. Banks, H. S. Beard, Y. Cao, A. E. Cho, W. Damm, R. Farid, A. K. Felts, T. A. Halgren, D. T. Mainz, J. R. Maple, R. Murphy, D. M. Philipp, M. P. Repasky, L. Y. Zhang, B. J. Berne, R. A. Friesner, E. Gallicchio and R. M. Levy, *J. Comput. Chem.*, 2005, **26**, 1752–1780.
[52] A. R. Leach, *Molecular Modelling: Principles and Applications*, Addison Wesley London Ltd, Harlow, UK, 1996.
[53] J. E. Lennard-Jones, *Proc. Royal Soc. Lond. A.*, 1924, **106**, 463–477.
[54] A. Rahman, *Phys. Rev.*, 1964, **136**, A405–A411.
[55] R. A. Buckingham, *Proc. Royal Soc. Lond. A.*, 1938, **168**, 264–283.
[56] P. M. Morse, *Phys. Rev.*, 1929, **34**, 57–64.
[57] C. A. Coulomb, *Histoire de l'Académie Royale des Sciences*, 1788, 569–577.
[58] C. A. Coulomb, *Histoire de l'Académie Royale des Sciences*, 1788, 578–611.
[59] J. Gumbart, L. G. Trabuco, E. Schreiner, E. Villa and K. Schulten, *Structure*, 2009, **17**, 1453–1464.
[60] D. Frenkel and B. Smit, *Understanding Molecular Simulation: From Algorithms to Applications*, Academic Press, San Diego, USA, 1996.
[61] P. P. Ewald, *Ann. Phys.*, 1921, **369**, 253–287.
[62] U. Essmann, L. Perera, M. L. Berkowitz, T. Darden, H. Lee and L. G. Pedersen, *J. Chem. Phys.*, 1995, **103**, 8577–8593.
[63] W. L. Jorgensen and J. Tirado-Rives, *J. Am. Chem. Soc.*, 1988, **110**, 1657–1666.
[64] K. Pluhackova and R. A. Böckmann, *J. Phys. Condens. Matter*, 2015, **27**, 323103.
[65] H. H. Rosenbrock, *Comput. J.*, 1960, **3**, 175–184.
[66] G. Wu, R. Mallipeddi and P. N. Suganthan, *Swarm Evol. Comput.*, 2019, **44**, 695–711.
[67] M. Wormington, C. Panaccione, K. M. Matney and D. K. Bowen, *Philos. Trans. R. Soc. London Ser. A*, 1999, **357**, 2827–2848.
[68] M. Björck, *J. Appl. Crystallogr.*, 2011, **44**, 1198–1204.
[69] M. Björck and G. Andersson, *J. Appl. Crystallogr.*, 2007, **40**, 1174–1178.
[70] F. Ott, *SimulReflec*, `http://www-llb.cea.fr/prism/programs/simulreflec/simulreflec.html`, (accessed 2019-03-04).
[71] P. A. Kienzle, M. Douchet, D. J. McGilivray, K. V. O'Donovan, N. K. Berk and C. F. Majkrzak, *NCNR Reflectometry Software*, `http://www.ncnr.nist.gov/reflpak`, (accessed 2018-03-04).
[72] R. Storn and K. Price, *J. Global Optim.*, 1997, **11**, 341–359.
[73] J. H. Holland, *Adaptation in Natural and Artificial Systems*, MIT Press, Boston, USA, 2nd edn., 1992.
[74] D. H. Ackley, *PhD*, University of Michigan, Michigan, US, 1987.
[75] J. Kennedy and R. Eberhart, Proceedings of ICNN'95, Perth, AU, 1995, pp. 1942–1948.

[76] Y. Shi and R. Eberhart, 1998 IEEE International Conference on Evolutionary Computation Proceedings, Anchorage, US, 1998, pp. 69–73.
[77] R. Poli, *J. Artfi. Evol. Appl.*, 2008, **2008**, 1–10.
[78] W. Krauth, *Statistical Mechanics: Algorithms and Computations*, Oxford University Press, Oxford, UK, 2006.
[79] D. S. Sivia and J. Skilling, *Data Analysis: A Bayesian Tutorial*, Oxford University Press, Oxford, 2nd edn., 2006.
[80] W. Gilks, S. Richardson and D. Spiegelhalter, *Markov Chain Monte Carlo in Practice*, CRC Press, Boca Raton, US, 1995.
[81] N. Metropolis, A. Rosenbluth, M. N. Rosenbluth, A. H. Teller and E. Teller, *J. Chem. Phys.*, 1953, **21**, 1087–1092.
[82] W. K. Hastings, *Biometrika*, 1970, **57**, 97–109.
[83] K. Levenberg, *Quart. Appl. Math.*, 1944, **2**, 164–168.
[84] D. W. Marquardt, *J. Soc. Indust. Appl. Math.*, 1963, **11**, 431–441.
[85] W. C. Swope, H. C. Andersen, P. H. Berens and K. R. Wilson, *J. Chem. Phys.*, 1982, **76**, 637–649.
[86] R. E. Rudd and J. Q. Broughton, *Phys. Rev. B*, 1998, **58**, R5893–R5896.
[87] E. Brini, E. A. Algaer, P. Ganguly, C. Li, F. Rodríguez-Ropero and N. F. A. van der Vegt, *Soft Matter*, 2013, **9**, 2108–2119.
[88] *RCSB PDB: Protein Data Bank*, `http://www.rcsb.org`, (accessed 2018-01-28).
[89] *Jmol: An Open-Source Java Viewer for Chemical Structures in 3D*, `http://www.jmol.org/`, (accessed 2018-01-28).
[90] L. Martínez, R. Andrade, E. G. Birgin and J. M. Martínez, *J. Comput. Chem.*, 2009, **30**, 2157–2164.
[91] H. C. Andersen, *J. Chem. Phys.*, 1980, **72**, 2384–2393.
[92] S. Nosé, *J. Chem. Phys.*, 1984, **81**, 511–519.
[93] H. J. C. Berendsen, J. P. M. Postma, W. F. van Gunsteren, A. DiNola and J. R. Haak, *J. Chem. Phys.*, 1984, **81**, 3684–3690.
[94] W. G. Hoover, *Phys. Rev. A*, 1985, **31**, 1695–1697.
[95] A. R. McCluskey, B. J. Morgan, K. J. Edler and S. C. Parker, *J. Open Source Educ.*, 2018, **1**, 19.
[96] A. R. McCluskey, B. J. Morgan, K. J. Edler and S. C. Parker, *Arm61/Pylj: Pylj-1.1.0*, `http://doi.org/10.5281/zenodo.1403828`, 2018.
[97] G. Bussi, D. Donadio and M. Parrinello, *J. Chem. Phys.*, 2007, **126**, 014101.
[98] M. Parrinello and A. Rahman, *J. Appl. Phys.*, 1981, **52**, 7182–7190.

3 Chemically consistent modelling of X-ray and neutron reflectometry

Abstract

This work presents the first example of the use of a chemically-consistent reflectometry model to co-refine X-ray reflectometry measurements at different surface pressures. This was coupled with a Markov chain Monte Carlo sampling methodology in order to rationalise the parameter inverse uncertainties and correlations. The chemically-consistent modelling approach was applied to the study of phospholipid monolayers at the air-deep eutectic solvent interface, which required that the head and tail group volume were not constrained. By co-refining multiple experimental datasets, it was possible to accurately model the experimental data without these constraints present.

Context

This project offers a severe method of coarse-graining for the analysis of neutron and X-ray reflectometry data. The system is coarse-grained to exist as a head group and a tail group of a phospholipid species. There is a chemical constraint present in the model, specifically that the number of head groups must be equal to the number of pairs of tail groups. However, there is no potential model considered beyond this "bonded" interaction. Additionally, this modelling approach is applied again in Chapter 4, as an example of the cutting edge of traditional modelling, against which the classical potential model-driven methods will be compared. The specific application of this modelling approach grew from a collaboration with experimental colleagues working on self-assembly in deep eutectic solvents. Therefore, this chemical system will be briefly introduced in Section 3.1. However, the main focus of this chapter will be the modelling methodology

3.1 Introduction

3.1.1 Deep eutectic solvents

Deep eutectic solvents (DES) are a class of green, sustainable liquids that may be obtained from the combination of ionic species with compounds capable of acting as hydrogen bond donors, such as sugar, alcohols, amines, and carboxylic acids [1, 2]. The resulting extensive hydrogen bonding network is capable of stabilising both species such that the eutectic mixture will remain liquid at room temperature [3–5]. Using different precursor materials can allow for the ability to tune the resulting solvent's physicochemical properties, such as polarity [6], viscosity and surface tension [1], network charge [7], and hydrophobicity [8, 9]. Recently DES have also been shown to exhibit a "solvophobic" effect through the promotion of surfactant micelle formation [10–13], phospholipid bilayer formation [14–16], and the ability to stability non-ionic polymer [17] and protein conformations [18].

Phospholipid monolayers at the air/water interface have been widely studied as simplistic models for biological membranes. As such, they have been used to gain insight into many biological processes that are technologically and medically relevant. For example, investigations at the air/salt-water interface have identified the importance that interactions between charged phospholipid head groups and ions present in solution have on the structure, monolayer packing and stability [19, 20]. However, the native environment for lipids in-vivo is far from simple aqueous solutions. In fact, it has been suggested [2, 4] that DES might form within the crowded cellular environment and could assist in solubilizing biological species in an intermediate environment between that of the hydrophobic phospholipid tail groups and the highly polar water-rich regions, thereby assisting survival under extreme conditions such as freezing temperatures and drought where the water content of the cells is restricted.

This chapter presents the first observation of phospholipid monolayer at an air-DES interface. Furthermore, this is one of a few examples of a phospholipid monolayer at the interface between air and a non-aqueous solvent, with only formamide noted previously [21]. Langmuir monol;ayers of non-phospholipidic surfactant molecules have also been noted at air-formamide and air-mercury interfaces [22–24]. In these works, the authors noted that the non-aqueous surface had an effect on the overall structure of the monolayer, but little was said about the underlying mechanism.

3.1.2 Optimisation and sampling in reflectometry analysis

The analysis of reflectometry data usually involves the use of some model-dependency. Therefore it is necessary to optimise the difference between our model, and the experimental dataset. An analytic method, such as the gradient descent method (Section 2.4.1) is not usually suitable for application to the optimisation of a reflectometry model, as these are only capable of optimisation to local minima, which would require accurate prior knowledge of the model structure [25]. Despite the analytic nature of the Maximum entropy (MaxEnt) optimisation method, this showed some success in the optimisation of reflectometry models, due in part to the immunity that optimisation carries to the production of a large number of solutions as it is able to determine the most likely model [26, 27]. This approach is computationally intensive and therefore become unusual to apply over other more efficient methods. Some aspects of the MaxEnt methods were replicated in the work of Sivia *et al.*, which employed Bayesian probability theory to rationalise the model selection [26, 28–30].

The use of analytic methods became less favoured as more frequently stochastic methods were used, as these offer a more pragmatic solution to the local minima problem. These are methods that utilise inherently random behaviour to determine a global minimum. The groove tracking method of Zhou and Chen [31, 32], was one of the first instances of a stochastic optimisation process applied to the analysis of reflectometry data. This randomly varied the scattering length density (SLD) of the layers in the model using a Monte Carlo approach. A similar approach used a simulated annealing approach, with a "temperature" factor that decreased as the number of iterations increases [33], however, this approach is still subject to the local minima problem as the probability of move acceptance decreased over time.

Both these Monte Carlo based approaches and the analytic methods previously discussed make the same perturbations to the fitted parameters during processing. However, this is the cause of the propensity to converge to a local minimum. This leads to the application of genetic algorithm derived methods for the optimisation of reflectometry models, beginning with the works of de Haan and Drijkoningen [34] and Dane *et al.* [35], who first applied this method to reflectometry analysis. These methods are designed to stochastically sample an entire given search-space, and therefore are more able to overcome the local minima issue and determine the vicinity of a global minimum. Following this initial application, genetic algorithms were used frequently in the optimisation of reflectometry models [36–39]. In particular, the work of Wormington *et al.* [39] showed the applicability of the differential evolution method towards reflectometry model optimisation, which resulted in the inclusion of such methods in many common reflectometry analysis software packages [40–45].

The use of Markov chain Monte Carlo (MCMC) methods to probe the probability distribution functions of the fitting parameters of a reflectometry model have grown in popularity [46–49]. This is due to the inclusion of MCMC methods in common analysis software packages such as Refl1D [45]. These methods enable the user to better understand the inverse uncertainties for the model. Additionally, they enable the quantification of the correlation between parameters, important in ensuring that the model applied is suitably constrained such as to reduce the cross-correlation present between parameters [42].

This work applies a differential algorithm [50, 51] to the optimisation of the reflectometry model. The search-space, available within the experimental uncertainty of the data is then sampled using MCMC, as implemented in `emcee` [52], to understand the parameter probability distributions and quantify the inter-parameter correlations.

3.1.3 Chemically-consistent modelling

The use of chemically-consistent modelling is common in the understanding of X-ray and neutron reflectometry measurements from phospholipid monolayers. The history of this modelling is introduced well by Campbell *et al.* [53]. While it is possible to model the neutron reflectometry from a phospholipid monolayer with a single layer model [54, 55], the use of at least two layers; representing the head and tail groups is more commonplace [56, 57]. Even when two layers are utilised, it is often the case that the volumes of the head and tail groups, V_i, are used as constraints in the modelling process, as the scattering length density (SLD) of a layer may be determined as follows,

$$\mathrm{SLD}_i = \frac{b_i}{V_i}(1 - \phi_i) + \mathrm{SLD}_s(\phi_s) \tag{3.1}$$

where, b_i is the scattering length of the head or tail, ϕ_i is the volume fraction of solvation by the solvent, SLD_s is the solvent scattering length density, and i indicates either the head or

tail layer. However, as noted by Campbell *et al.* [53], this method often fails to account for the compaction of the carbon chains under elevated surface pressures [58, 59], which may lead to a volume reduction of up to $\sim 15\,\%$. Furthermore, as discussed in Section 3.3, the use of a constrained head group volume may also impact the modelling process in situations where the volume is poorly defined.

Equation 3.1 enables the use of chemical-inference in the modelling approach for reflectometry data. This allows for the co-refinement of neutron reflectometry data where different isotopic-contrasts of the phospholipid species or solvent have been used. This is possible based on the expectation that the effect of contrast variation on the structure and chemistry of the monolayer will be negligible, and therefore the same values of all parameters in the fitting, except b_i and SLD_s may be constrained between the different measurements [60]. In this work, similar logic was applied, with the assumption that the volume of the head and tail groups is constant across different surface pressures, while the lipid phase is the same. This means that, for this system, all of the fitted parameters may be constrained aside from the tail thickness, head solvation, and interfacial roughness, across the different surface pressure measurements. This is the first time that such a methodology has been applied to the analysis of XRR and NR, additionally, it is believed that this ability to co-refine XRR measurements enables a greater understanding of the structure than that possible from a single measurement.

3.2 Experimental

The experimental measurements were designed and conducted by Drs Tom Arnold, Andrew Jackson, Adrian Sanchez-Fernandez, and Prof. Karen Edler, with the assistance of Dr Richard Campbell. My role in this work was entirely on the analysis of the measurements and the development of the chemically-consistent model. However, it is necessary to briefly discuss the materials and experimental methods used to enable a complete understanding of the context of the work.

3.2.1 Materials

Choline chloride ($99\,\%$, Sigma-Aldrich), glycerol ($99\,\%$, Sigma-Aldrich), $\mathrm{d_9}$-choline chloride ($99\,\%$, $98\,\%$ D, CK Isotopes), and $\mathrm{d_8}$-glycerol ($99\,\%$, $98\,\%$ D, CK Isotopes) were used in the preparation of the deep eutectic solvent (DES). This is achieved by mixing a 1:2 ratio of choline-chloride and glycerol and heating at $80\,^{\circ}\mathrm{C}$ until a homogeneous, transparent liquid is formed [1]. This was then stored under a dry atmosphere to reduce the amount of water dissolved in the solvent.

The limited availability of deuterated precursors lead to only a fully protonated (hDES) and a partially deuterated (hdDES) being prepared and used in the neutron reflectometry measurements. The partially deuterated subphase was prepared using the following mixtures of precursors: $1\,\mathrm{mol}$ of $0.38\,\mathrm{mol}$ fraction of h-choline-chloride/$0.62\,\mathrm{mol}$ fraction of d-choline-chloride; and $2\,\mathrm{mol}$ of $0.56\,\mathrm{mol}$ fraction of h-glycerol/$0.44\,\mathrm{mol}$ fraction of d-glycerol.

The water content of the DES was assessed before and after each experiment by Karl-Fisher titration (Mettler Toledo DL32 Karl-Fischer Coulometer, Aqualine Electrolyte A, Aqualine Catholyte CG A) and found to be always below $0.3\,\mathrm{wt}/\%$. This was taken to be a negligible amount and would not have a considerable impact on the DES characteristics [3, 4].

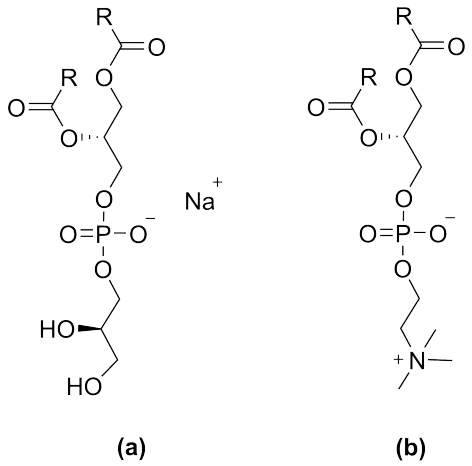


FIGURE 3.1: The two phospholipid forms investgated in this work, where R indicates the hydorcarbon tail; (a) phosphatidylglycerol (PG), (b) phosphocholine (PC).

1,2-dipalmitoyl-*sn*-glycero-3-phosphocholine (DPPC, C_{16} tails, $>99\,\%$), 1,2-dimyristoyl-*sn*-glycero-3-phosphocholine (DMPC, C_{14} tails, $>99\,\%$), and the sodium salt of 1,2-dimyristoyl-*sn*-glycero-3-phospho-(1'-rac-glycerol) (DMPG, C_{14} tails, $>99\,\%$) were obtained from Avanti Polar Lipids and 2-dilauroyl-*sn*-glycero-3-phosphocholine (DLPC, C_{12} tails, $>99\,\%$) was obtained from Sigma Aldrich and all were used without further purification. Deuterated versions of DPPC (d_{62}-DPPC, $>99\,\%$, deuterated tails-only) and DMPC (d_{54}-DMPC, $>99\,\%$, deuterated tails-only) were obtained from Avanti Polar Lipids and used without further purification. These lipids were dissolved in chloroform solution ($0.5\,\mathrm{mg\,mL^{-1}}$) at room temperature. PC indicates where the phospholipid molecule contains a phosphocholine head group, while PG indicates a phosphatidylglycerol head group, the chemical structures of these can be seen in Figure 3.1.

In the X-ray reflectometry (XRR) experiment, the sample was prepared in-situ using the standard method for the spreading of insoluble monolayers on water: a certain amount of the phospholipid solution was spread on the liquid surface. Following the evaporation of the chloroform, it is assumed that the resulting system is a subsurface of solvent with a monolayer of the phospholipid at the interface. The surface concentration is then controlled by opening and closing the PFTE barriers of a Langmuir trough. To reduce the volumes used in the neutron reflectometry (NR) experiments, a small Delrin adsorption trough was used that did not have controllable barriers. Therefore, although the surface concentration was nominally the same as for the XRR, the lack of precise control meant that it was determined to be inappropriate to co-refine the XRR and NR contrasts together.

3.2.2 Methods

The XRR measurements were carried out at the I07 beamline at the Diamond Light Source, with a photon energy of $12.5\,\mathrm{keV}$ using the double-crystal deflector system [61]. The reflected intensity was measured for a q range of 0.018 to $0.7\,\mathrm{\AA}^{-1}$. The data were normalised with respect to the incident beam and the background was measured from off-specular reflection and subsequently subtracted. All of the samples were allowed at least one hour to equilibrate and preserved under an argon atmosphere. XRR data were collected for each of the phospholipids, DLPC, DMPC, DPPC, and DMPG at four surface pressures each (DLPC: 20, 25, 30 and $35\,\mathrm{mN\,m^{-1}}$, DMPC: 20, 25, 30 and $40\,\mathrm{mN\,m^{-1}}$, DPPC: 15, 20, 25 and $30\,\mathrm{mN\,m^{-1}}$,

DMPG: 15, 20, 25 and 30 $\mathrm{mN\,m^{-1}}$), as measured with an aluminium Wilhelmy plate; measurements were conducted at 7 and 22 °C. An aluminium Wilhelmy plate was used over a traditional paper due to the low wettability of paper by the DES.

The NR experiments were performed on the FIGARO instrument at the Institut Laue-Langevin using time-of-flight methods [62]. Data were collected at two incident angles; 0.62 and 3.8°, providing a q range from 0.005 to 0.18 $\mathrm{Å^{-1}}$. Two surface pressures for each phospholipid and contrast were measured (DMPC: 20 and 25 $\mathrm{mN\,m^{-1}}$, DPPC: 15 and 20 $\mathrm{mN\,m^{-1}}$). As with the XRR measurements, the samples were given at least one hour to equilibrate, kept under an inert atmosphere. All measurements were conducted at 22 °C.

3.3 Data analysis

XRR and NR methods have a well documented history for the analysis of the structure of phospholipid monolayers at the air-water interface [19, 20, 63–67]. Typically these have involved using a model-dependent analysis method, however, the modelling approaches have varied significantly in the number of layers used, the shape of the layers, the use of interfacial roughness, the parameterisation of constraints employed, and even the method by which the reflectometry profile was calculated from the model. Recently, an evaluation of the applicability of different models for surfactant and phospholipid monolayers using NR outlined a view of "best practice" [53]. However, frequently the constraints employed in the modelling process include the head and tail volume for the phospholipid head and tail groups. These values are taken from a variety of other techniques, some examples are shown in Table 3.1.

Table 3.1 provides a general consensus that the volume of the phosphocholine head group (PC) is 320 to 360 $\mathrm{Å^3}$, while the phosphatidylglycerol head group (PG) is 289 to 291 $\mathrm{Å^3}$. However, these values were all determined from experiments or simulations where the head group was interacting with water molecules. It is not clear if this will influence the volume that it occupies, and if that volume will change in the presence of a non-aqueous solvent, such as the DES considered herein. The charged nature of the zwitterionic and anionic phospholipid head groups may have different interactions with the polar, but neutral water and the charged DES [74]. Additionally, it is known that, on water, increased surface pressures and the associated Liquid-Expanded to Liquid-Condensed phase transition will lead to a compression of the phospholipid tail volume, compared to the values in Table 3.1 [59, 75], and that this compaction has not necessarily been accounted for in the literature [53].

These factors meant that it was necessary to develop a model that was considerate of the phospholipid chemistry while applying as much of the "best practice" from Campbell *et al.* [53] as possible, and ensuring that the head and tail group volumes were not constrained parameters. The lack of having these normally constrained parameters meant that it was necessary to consider methods by which the reflectometry measurements could be co-refined, in a similar fashion to contrast variation co-refinement in neutron reflectometry. This could be achieved by the co-refinement of reflectometry measurements at different surface pressures, as the model was considerate of the phospholipid chemistry, and the different surface pressures were in the same phase; Liquid-Condensed (LC) for DPPC and Liquid-Expanded (LE) for DMPC, DLPC, and DMPG. Therefore the head and tail group volumes will remain constant, and only the surface concentration and normal tail thickness will vary.

The chemically-constrained model that has been used in this work was implemented in the Python library `refnx` [43, 76]. The software enables the inclusion of custom model

TABLE 3.1: Lipid component volumes extracted from different literature sources. V_l corresponds to the total phospholipid volume, V_t to the tail group volume, V_h to the head group volume, MD to molecular dynamics simulations, WAXS to wide-angle X-ray scattering, NB to neutral buoyancy, and DVTD to differential vibrating tube densimetry.

Phospholipid	DPPC			DMPC		DLPC		DMPG	POPG
Ref.	[68]	[69]	[70, 71][a]	[68]	[70, 71]	[68]	[70, 71]	[72]	[73]
V_l/Å^3	1287.3 ± 25.5	1148 ± 2	1268.2 ± 32.1	1172.5 ± 25.1	1155.4 ± 30.0	1057.7 ± 24.7	1046.6 ± 28.0	1011.4	1203
V_t/Å^3	966.4 ± 5.4	829 ± 4	924.7 ± 17.6	851.5 ± 5.0	815.9 ± 15.5	736.8 ± 4.6	707.1 ± 13.5	720.4	914
V_h/Å^3	320.9 ± 20.1	319 ± 6	339.5 ± 14.5	320.9 ± 20.1	339.5 ± 14.5	320.9 ± 20.1	339.5 ± 14.5	291.0	289
Method	MD	WAXS	NB	MD	NB	MD	NB	DVTD	MD
T/°C	50	24	30	50	30	50	30	20	25

[a]The values for the head component in Kučerka *et al.* [70], were taken from Balgavý *et al.* [71]

TABLE 3.2: The invarient parameters within the chemically-consistent model.

Component	b_t/fm	b_h/fm	t_t/Å	SLD/10^{-6} Å^{-2}
X-ray				
DPPC	6827	4635	20.5[1]	–
DMPC	5924	4635	18.0[1]	–
DLPC	5021	4635	15.5[1]	–
DMPG	5924	4694	18.0[1]	–
Air	–	–	–	0
DES	–	–	–	10.8[2]
Neutron				
d_{54}-DMPC			18.0[1]	–
d_{62}-DPPC			20.5[1]	–
h-DES	–	–	–	0.43[2]
hd-DES	–	–	–	3.15[2]

classes that feed parameters into the Abelès model (Section 2.2.6) [77, 78]. Our chemically-consistent model class can be seen in Code Block 3.1, and is shared under a CC BY-SA 4.0 license in the ESI for the associated publication [79]. In order to ensure that the phospholipid chemistry was consistent both within the phospholipid molecule and across the different surface pressures, Code Block 3.2 was implemented.

The chemically-consistent model, that is outlined in Code Block 3.1, consisted of two layers that define the phospholipid monolayer; the head layer at the interface with the solvent and the tail layer at the air interface. The head groups have a scattering length that can be calculated from a summation of the X-ray or neutron atomic scattering lengths, b_h, and a volume, V_h. These groups make up a layer of a given thickness, d_h, which has some interfacial roughness, σ_h, within which some volume fraction of solvent may penetrate, ϕ_h. The tail layer is defined in the same way, however, the tail thickness, d_t, is constrained such that it can be no greater than the maximum extended length for the phospholipid tail (defined as the Tanford length, t_t [80]), which is given in Table 3.2, and that no solvent may penetrate into the layer (e.g. $\phi_t = 0$). Therefore, the SLD may be determined as discussed in Equation 3.1. Based on the work of Campbell *et al.* [53], a single value for the interfacial roughness was fitted for all of the interfaces, including the subphase (i.e. $\sigma_t = \sigma_h = \sigma_s$), as there is only a single phospholipid molecule type present in each monolayer. Therefore, any capillary wave roughness at the air-DES interface is carried monotonically through the layers. The interfacial roughness was constrained to be greater than 3.3 Å, in agreement with previous work [10].

The constraints implemented in Code Block 3.2 involved two aspects. The first was to ensure that the number density of head groups and pairs of tail groups was kept the same. This was achieved with the following relation [81],

$$\phi_h = 1 - \left(\frac{d_t V_h}{V_t d_h}\right). \tag{3.2}$$

The second aspect was to enforce chemically-consistant constraints across the measurements that were conducted at different surface pressures. This was achieved by constraining the head and tail group volumes and the head layer thickness such that they do not vary between the different surface pressure measurements.

[1] Values obtained from the Tanford formula [80].
[2] Values obtained from Sanchez-Fernandez *et al.* [10].

CODE BLOCK 3.1: The chemically-consistent model class that was implemented in `refnx` [43, 76].

```
import numpy as np
from refnx.analysis import possibly_create_parameter as pcp
from refnx.analysis import Parameters
from refnx.reflect import Component


class VolMono(Component):
    def __init__(self, vol, b, d_h, c_length, name=""):
        """
        The custom model class to enable control of the
        phospholipid head and tail volumes.

        Parameters
        ----------
        vol: float, array_like
            The initial values for the head and tail group volumes.
        b: complex, array_like
            The calculated scattering length for the head and tail
            groups.
        d_h: float
            The initial value for the thickness of the head group
            region.
        c_length: int
            Number of carbon atoms in the phospholipid tail.
        name: str
            A name to be given to the object.
        """
        super(VolMono, self).__init__()
        t_t = 1.54 + 1.265 * c_length
        self.vol = [
            pcp(vol[0], "{}-V_h".format(name)),
            pcp(vol[1], "{}-V_t".format(name)),
        ]
        self.realb = [
            pcp(b[0].real, name="{}-b_h".format(name)),
            pcp(b[1].real, name="{}-b_t".format(name)),
        ]
        self.imagb = [
            pcp(b[0].imag, name="{}-ib_h".format(name)),
            pcp(b[1].imag, name="{}-ib_t".format(name)),
        ]
        self.d = [
            pcp(d_h, name="{}-d_h".format(name)),
            pcp(t_t * 0.8, name="{}-d_t".format(name)),
        ]
        self.phi = [
            pcp(0.5, name="{}-phi_h".format(name)),
            pcp(0.0, name="{}-phi_t".format(name)),
        ]
        self.sigma = pcp(3.0, name="{}-sigma".format(name))
        self.name = name

    @property
    def slabs(self):
        """
        Returns
        -------
        float, array_like
            Slab representaions of phospholipid monolayer.
        """
        layers = np.zeros((2, 5))
        layers[0, 0] = self.d[1]
        layers[0, 1] = self.realb[1] * 1.0e16 / self.vol[1]
        layers[0, 2] = self.imagb[1] * 1.0e16 / self.vol[1]
        layers[0, 3] = self.sigma
        layers[0, 4] = self.phi[1]
        layers[1, 0] = self.d[0]
        layers[1, 1] = self.realb[0] * 1.0e16 / self.vol[0]
        layers[1, 2] = self.imagb[0] * 1.0e16 / self.vol[0]
        layers[1, 3] = self.sigma
        layers[1, 4] = self.phi[0]
        return layers
```

CODE BLOCK 3.2: The `set_constraints` that was used to impose chemical-consistency on the phospholipid monolayer structure..

```
def set_constraints(
    lipids,
    structures,
    hold_tails=False,
    hold_rough=False,
    hold_phih=False,
):
    i = 0
    lipids[i].phi[0].constraint = 1 - (
        lipids[0].vol[0] / lipids[0].vol[1]
    ) * (lipids[i].d[1] / lipids[i].d[0])
    lipids[i].sigma.constraint = structures[i][-1].rough
    for i in range(1, len(lipids)):
        lipids[i].vol[0].constraint = lipids[0].vol[0]
        lipids[i].vol[1].constraint = lipids[0].vol[1]
        lipids[i].d[0].constraint = lipids[0].d[0]
        if hold_tails:
            lipids[i].d[1].constraint = lipids[0].d[1]
```

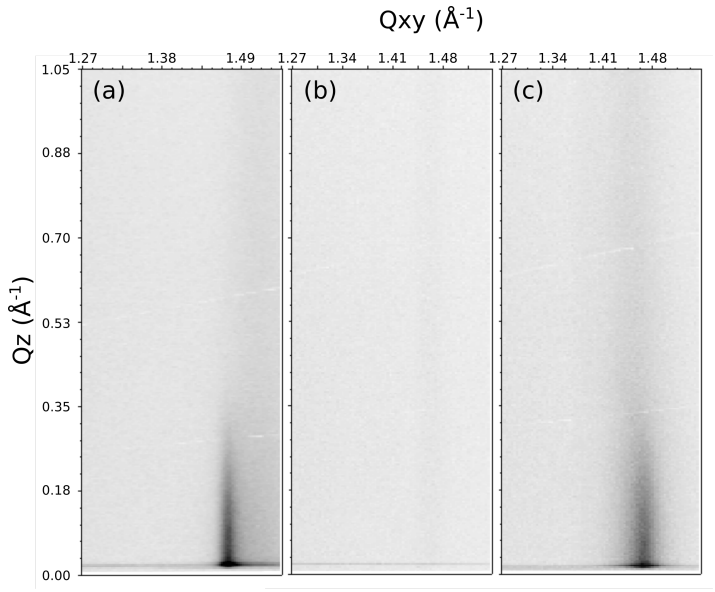


FIGURE 3.2: The GIXD patterns; (a) DPPC at $30\,\mathrm{mN\,m^{-1}}$ and $22\,^{\circ}\mathrm{C}$, (b) DMPC at $30\,\mathrm{mN\,m^{-1}}$ and $22\,^{\circ}\mathrm{C}$, and (c) DMPC at $30\,\mathrm{mN\,m^{-1}}$ and $7\,^{\circ}\mathrm{C}$.

The justification for constraining the tail volume is built on the assumption that the phospholipids remain in the same phase. On water, this may be demonstrated with a Langmuir isotherm. However, it was not possible to collect consistent Langmuir isotherm measurements, due to the high viscosity of the DES. Instead, grazing incidence X-ray diffraction was used to confirm the phases of DMPC and DPPC at $30\,\mathrm{mN\,m^{-1}}$. Figure 3.2 shows the GIXD data from different phospholipids at different temperatures. Unfortunately, all the patterns show a weak artefact due to scattering from the Teflon trough. However, there are clear (2, 0) diffraction peaks in the GIXD pattern for DPPC at $22\,^{\circ}\mathrm{C}$ and DMPC at $7\,^{\circ}\mathrm{C}$ indicating that both lipids are in the LC phase. This peak was also present at other surface pressures (data not shown). The peak position corresponded well with that found for DPPC in water [82]. DMPC at $22\,^{\circ}\mathrm{C}$ showed no evidence of a diffraction peak indicating the presence of the LE phase. It was assumed that DLPC and DMPG were also in the LE phase as there is no reason for the phase behaviour of these systems to differ significantly from that of DMPC at room temperature.

Initially, this chemically-consistent modelling approach was applied only to the XRR data. The tail layer thickness and interfacial roughness were allowed to vary independently across

the surface pressures, while the other parameters were constrained as discussed above or held constant to the values given in Table 3.2. For each co-refinement of four XRR measurements, there were, in total, eleven degrees of freedom. Throughout all of the analyses, the intensity scale factor was allowed to vary freely, while the background was constrained to the intensity at the largest q-value.

Following this, the head and tail group volumes, and the head layer thickness that were found from the XRR analysis were used as fixed variables for the refinement of the NR measurements. The reduced the number of fitted parameters in the NR data to two, namely the thickness of the tail layer, d_t, and the interfacial roughness, $\sigma_{t,h,s}$, for the co-refinement of two datasets. Table 3.2 also presents details of the scattering lengths and SLDs using for the NR refinement. Again, the intensity scale factor was allowed to vary freely and the background constrained to the intensity at the largest q-value.

In both the XRR and the NR analysis, the refinement of the chemically-consistent model to the experimental data involved the transformation of the reflectometry calculated from the model and the data into Rq^4-space, such that the contribution of the Fresnel decay was removed [83]. The model was then optimised using the differential evolution method that is available within the `scipy` library [51]. This refined the parameters to give the best fit to the data. Markov chain Monte Carlo (MCMC) was then used to probe the search-space available to each parameter, given the experimental uncertainty of the data. The MCMC sampling method used was Goodman & Weare's Affine Invarient Ensemble [84] as impliemented in the `emcee` package [52]. This enabled the determination of the probability distribution for each of the parameters, and therefore the quantification of their inverse uncertainty, given the uncertainty in the experimental data. A Shapiro-Wilk test [85] was used to determine the normality of each of the probability distribution functions (PDFs). If the PDF failed the test the value was quoted with asymmetric confidence intervals, compared with the symmetric confidence intervals given for those that passed the Shapiro-Wilk test. It is important to note that the PDFs and therefore the determined confidence intervals are not true confidence intervals, and account only for the uncertainty that is present in the data, i.e. they do not account for systematic uncertainty in the measurement technique. In addition to determining parameter confidence intervals, it was also possible to use these probability distributions to understand the correlations present between the parameters and the impact this has on the fitting process. The correlation was quantified using the Pearson correlation coefficient [86], a common statistical definition for the level of correlation present between two variables. The Pearson correlation coefficient can have values that range from -1 to 1, with a value of -1 corresponding to a complete negative correlation (an increase in one variable is associated with a decrease in the other), while a value of 1 corresponds to a complete positive correlation (an increase in one variable is associated with a similar increase in the other), a value of 0 indicates no correlation between the two variables. The MCMC sampling involved 200 walkers that were used for 1000 iterations, following a burn-in of 200 iterations.

3.4 Results & Discussion

3.4.1 X-ray reflectometry

The chemically-consistent model was co-refined across XRR measurements at all four surface pressures for each phospholipid. The resulting XRR profiles and associated SLD profiles are shown in Figure 3.3. Tables 3.3, 3.4, 3.5, and 3.6 gives the parameters for each of the

TABLE 3.3: The best-fit values, and associated $95\,\%$ confidence intervals for each of the varying parameters for each phospholipid at the lowest surface pressure measured from XRR. The values of ϕ_h were obtained from the appropriate use of Equation 3.2.

Phospholipid Surface Pressure/$\mathrm{mN\,m^{-1}}$	DPPC 15	DMPC 20	DLPC 20	DMPG 15
V_t/Å^3	765.31 ± 0.38	718.76 ± 0.54	625.50 ± 3.70	734.00 ± 0.58
V_h/Å^3	322.00 ± 0.24	339.53 ± 0.27	331.38 ± 0.81	329.96 ± 0.34
d_h/Å	$12.70^{+0.03}_{-0.03}$	$13.21^{+0.04}_{-0.04}$	11.00 ± 0.13	13.95 ± 0.03
d_tÅ	15.95 ± 0.01	9.39 ± 0.01	6.95 ± 0.03	5.70 ± 0.05
$\sigma_{t,h,s}$/Å	3.88 ± 0.00	4.20 ± 0.01	4.22 ± 0.03	4.65 ± 0.01
$\phi_h/\times 10^{-2}$	47.14 ± 0.21	66.41 ± 0.17	66.53 ± 0.76	81.62 ± 0.18

TABLE 3.4: The best-fit values, and associated $95\,\%$ confidence intervals for each of the varying parameters for each phospholipid at the second lowest surface pressure measured from XRR. The values of ϕ_h were obtained from the appropriate use of Equation 3.2.

Phospholipid Surface Pressure/$\mathrm{mN\,m^{-1}}$	DPPC 20	DMPC 25	DLPC 25	DMPG 20
V_t/Å^3	765.31 ± 0.38	718.76 ± 0.54	625.50 ± 3.70	734.00 ± 0.58
V_h/Å^3	322.00 ± 0.24	339.53 ± 0.27	331.38 ± 0.81	329.96 ± 0.34
d_h/Å	$12.70^{+0.03}_{-0.03}$	$13.21^{+0.04}_{-0.04}$	11.00 ± 0.13	13.95 ± 0.03
d_tÅ	$16.57^{+0.01}_{-0.01}$	12.12 ± 0.01	8.29 ± 0.05	10.64 ± 0.01
$\sigma_{t,h,s}$/Å	4.09 ± 0.00	3.92 ± 0.01	$4.36^{+0.02}_{-0.03}$	3.94 ± 0.01
$\phi_h/\times 10^{-2}$	45.07 ± 0.21	56.66 ± 0.22	60.07 ± 0.96	65.71 ± 0.14

phospholipids at each surface pressure measured, as well as the details fo ϕ_h, as determined from Equation 3.2.

Following the structural determination of the monolayer from the XRR measurements. NR was used to confirm the values of the head and tail group volumes that had been determined. The resulting NR profiles and associated SLD profiles, at both surface pressures measured can be found in Figure 3.4. Table 3.7 gives the parameters as determined from the NR measurements, along with ϕ_h as determined from Equation 3.2.

3.4.2 Effect of compression on the monolayer thickness

From Tables 3.3, 3.4, 3.5, 3.6, and 3.7, we can see that, as expected and shown in previous work [19, 87], the thickness of the tail layer increases as the number of carbon atoms in the tail chain increases. Furthermore, the thickness of the tail layers determined here agrees well with values found for water-analoguyes; 13.72 ± 0.01 Å at $30\,\mathrm{mN\,m^{-1}}$ in DES compared with 15.8 Å at $30\,\mathrm{mN\,m^{-1}}$ in water for DMPC, and 16.91 ± 0.01 Å at $30\,\mathrm{mN\,m^{-1}}$ in DES compared with 16.7 Å at $40\,\mathrm{mN\,m^{-1}}$ in water for DPPC.

The variation of the tail layer thickness in the models with surface pressure is given for each phospholipid in Figure 3.5. For all of the phospholipids, as the surface pressure increases, the thickness of the tail layer also increases to a point before plateauing; for DPPC this occurs at $20\,\mathrm{mN\,m^{-1}}$, DMPC at $30\,\mathrm{mN\,m^{-1}}$, and for DMPG and DLPC can be assumed to be

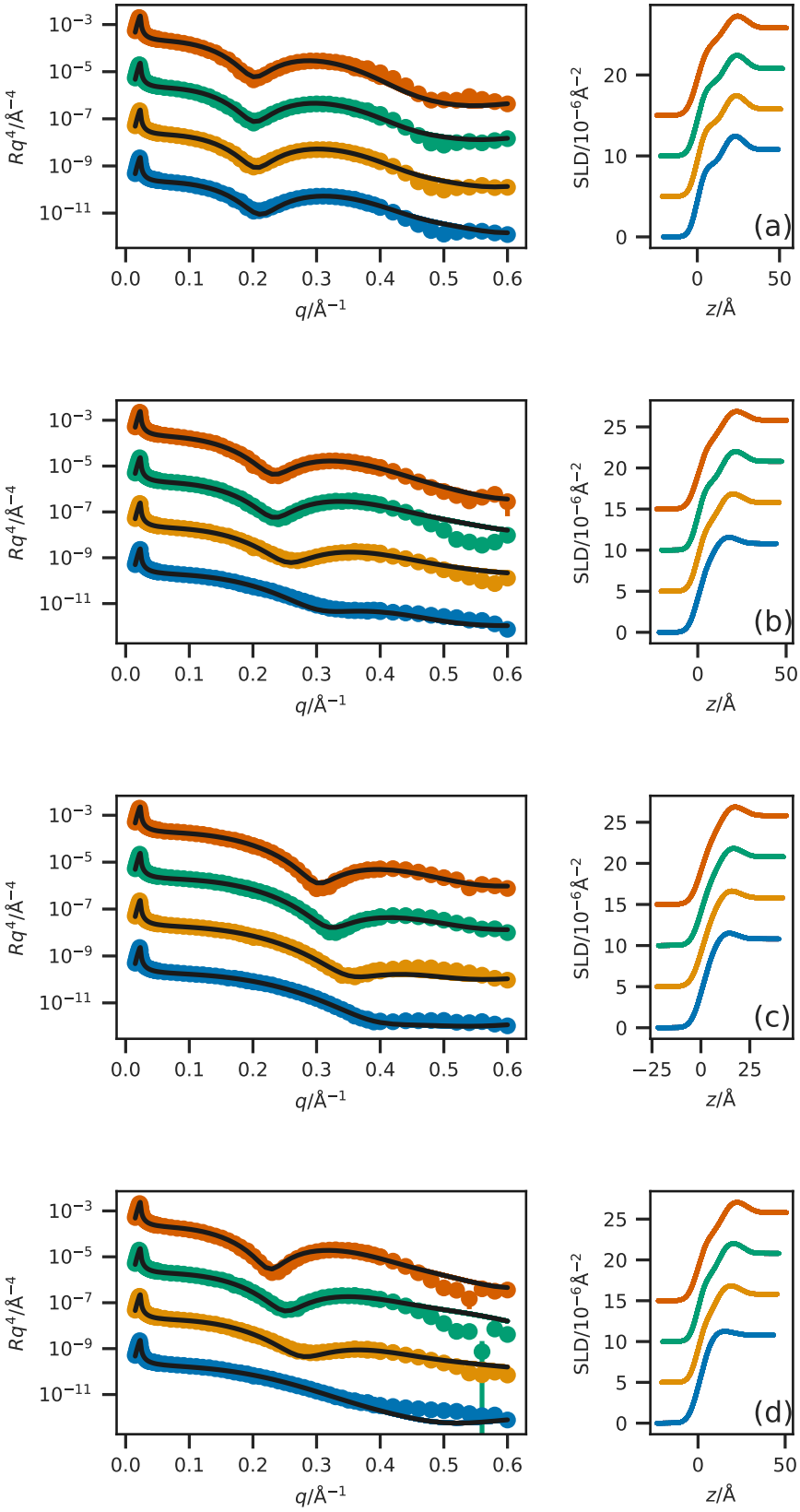


FIGURE 3.3: The XRR profiles (left) and SLD profiles (right) for each of the four phospholipids; (a) DPPC, (b) DMPC, (c) DLPC, and (d) DMPG, at the four measured surface pressures; increasing in surface pressure from blue, orange, green to red. The different surface pressure XRR profiles have been offset in the y-axis by two orders of magnitude and the SLD profiles offset in the y-axis by 5×10^{-6} Å^{-2}, for clarity.

TABLE 3.5: The best-fit values, and associated 95 % confidence intervals for each of the varying parameters for each phospholipid at the second highest surface pressure measured from XRR. The values of ϕ_h were obtained from the appropriate use of Equation 3.2.

Phospholipid Surface Pressure/$\mathrm{mN\,m^{-1}}$	DPPC 25	DMPC 30	DLPC 30	DMPG 25
V_t/Å^3	765.31 ± 0.38	718.76 ± 0.54	625.50 ± 3.70	734.00 ± 0.58
V_h/Å^3	322.00 ± 0.24	339.53 ± 0.27	331.38 ± 0.81	329.96 ± 0.34
d_h/Å	$12.70^{+0.03}_{-0.03}$	$13.21^{+0.04}_{-0.04}$	11.00 ± 0.13	13.95 ± 0.03
d_tÅ	16.83 ± 0.01	13.72 ± 0.01	9.51 ± 0.06	12.24 ± 0.01
$\sigma_{t,h,s}$/Å	4.31 ± 0.00	3.86 ± 0.00	4.16 ± 0.04	3.81 ± 0.00
$\phi_h/\times 10^{-2}$	44.23 ± 0.22	50.94 ± 0.25	54.20 ± 1.10	60.57 ± 0.17

TABLE 3.6: The best-fit values, and associated 95 % confidence intervals for each of the varying parameters for each phospholipid at the highest surface pressure measured from XRR. The values of ϕ_h were obtained from the appropriate use of Equation 3.2.

Phospholipid Surface Pressure/$\mathrm{mN\,m^{-1}}$	DPPC 30	DMPC 40	DLPC 35	DMPG 30
V_t/Å^3	765.31 ± 0.38	718.76 ± 0.54	625.50 ± 3.70	734.00 ± 0.58
V_h/Å^3	322.00 ± 0.24	339.53 ± 0.27	331.38 ± 0.81	329.96 ± 0.34
d_h/Å	$12.70^{+0.03}_{-0.03}$	$13.21^{+0.04}_{-0.04}$	11.00 ± 0.13	13.95 ± 0.03
d_tÅ	16.91 ± 0.01	13.82 ± 0.01	10.30 ± 0.07	13.99 ± 0.01
$\sigma_{t,h,s}$/Å	4.90 ± 0.00	4.53 ± 0.01	4.35 ± 0.04	4.44 ± 0.00
$\phi_h/\times 10^{-2}$	43.95 ± 0.22	50.58 ± 0.25	50.40 ± 1.19	54.91 ± 0.19

TABLE 3.7: The best-fit values, and associated 95 % confidence intervals for each of the varying parameters for each phospholipid at the each surface pressure from NR. The values of ϕ_h were obtained from the appropriate use of Equation 3.2.

Phospholipid Surface Pressure/$\mathrm{mN\,m^{-1}}$	DPPC 15	DPPC 20	DMPC 20	DMPC 25
d_tÅ	12.67 ± 0.13	15.43 ± 0.08	14.81 ± 0.13	$17.98^{+0.01}_{-0.03}$
$\sigma_{t,h,s}$/Å	4.77 ± 0.16	$3.31^{+0.04}_{-0.01}$	3.47 ± 0.15	$3.30^{+0.02}_{-0.00}$
$\phi_h/\times 10^{-2}$	58.04 ± 0.44	48.86 ± 0.26	47.06 ± 0.48	$35.71^{+0.11}_{-0.03}$

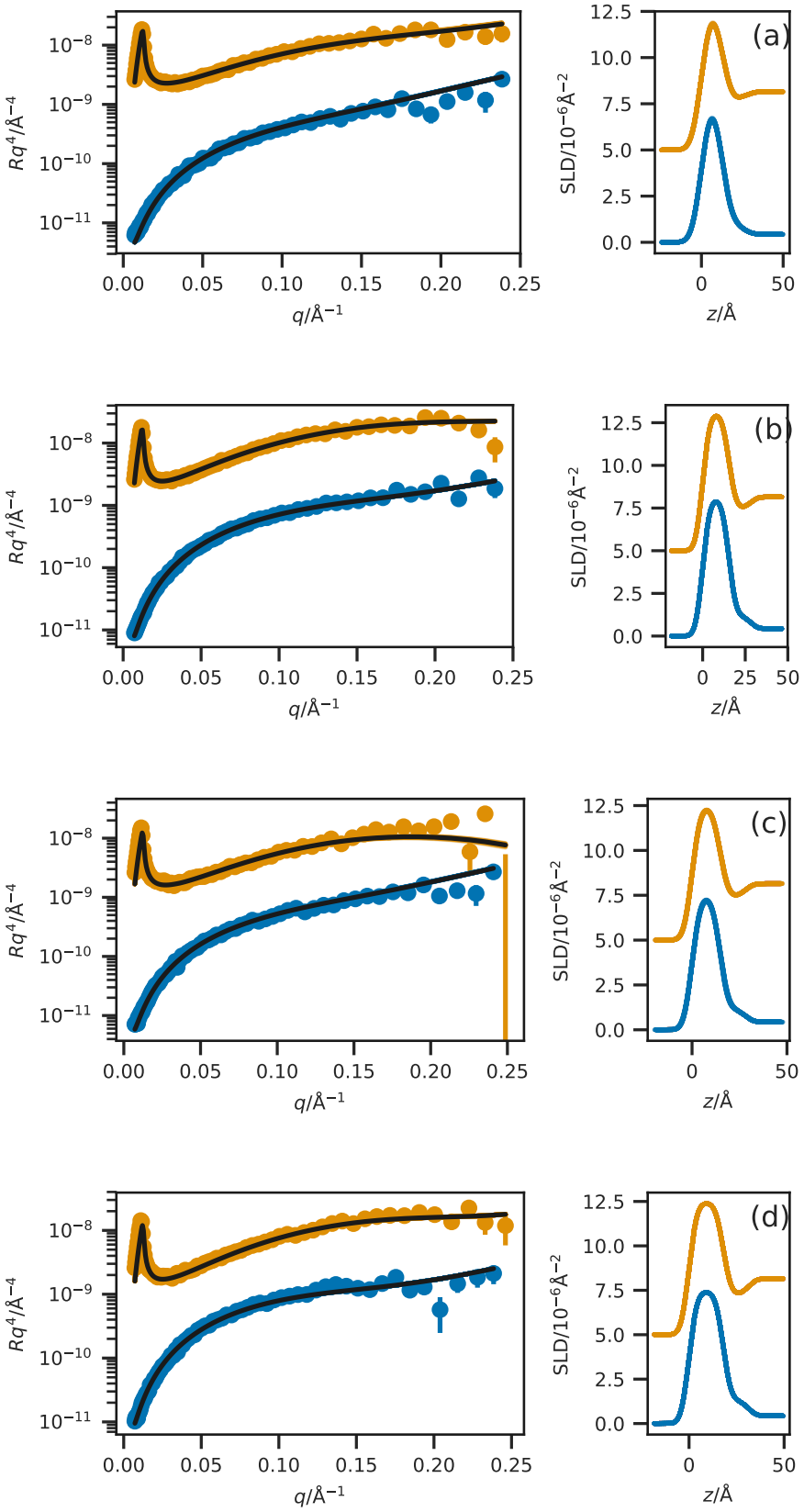


FIGURE 3.4: The NR profiles (left) and SLD profiles (right) for each of the four phospholipids; (a) DPPC at $15\,\text{mN}\,\text{m}^{-1}$, (b) DPPC at $20\,\text{mN}\,\text{m}^{-1}$, (c) DMPC at $20\,\text{mN}\,\text{m}^{-1}$, and (d) DMPC at $25\,\text{mN}\,\text{m}^{-1}$, where the blue data indicates the h-DES contrast, while the orange is the hd-DES. The different surface pressure XRR profiles have been offset in the y-axis by an order of magnitude and the SLD profiles offset in the y-axis by $5 \times 10^{-6}\,\text{Å}^{-2}$, for clarity.

at higher pressures than those studied. This relationship of increasing tail layer thickness with increasing surface pressure has been noted previously for DMPC [63] and DPPC [53] at the air-water interface. This can be easily understood as the angle of the tail group with respect to the surface normal decreasing as the surface pressure increases.

3.4.3 Effect of compressure on solvent fraction

In Figure 3.5, it is clear that for all four phospholipids, as the surface pressure is increased there is a corresponding reduction in the volume fraction of solvent in the phospholipid head layer, This can be rationalised by considering that when the surface pressure is increased, there is a corresponding increase in surface concentration, hence the free volume available to the solvent is less. A similar effect has been observed when increase the surface pressure from $11\,\mathrm{mN\,m^{-1}}$ to $31\,\mathrm{mN\,m^{-1}}$ for a mixed DMPC/DMPG monolayer at the air-water interface [63].

3.4.4 Effect of compression on the lipid tail component volumes

It can be seen from comparing Table 3.1 with Tables 3.3, 3.4, 3.5, and 3.6 that the volume of the phospholipid tails are significantly lower in the current measurement than found previously, by other techniques. It is unlikely that this is a result of the DES subphase, due to the hydrophobic nature of these tail groups. However, this reduction has been shown previously [53], where it was rationalised by the compaction of the monolayer at elevated surface pressure. In that work, the optimal value for the tail group volume of DPPC was found to be $772\,\text{Å}^3$ at a surface pressure of $35\,\mathrm{mN\,m^{-1}}$, which agree well with the value of 765.31 ± 0.38 Å^3 found in this work at surface pressures of 15, 20, 25 and $30\,\mathrm{mN\,m^{-1}}$. This reduction was found to be between 8 to $12\,\%$ for DPPC, DMPC, DLPC when compared with the literature sources at 24 and $30\,^{\circ}\mathrm{C}$. This is close to the maximum compression percentage of $15\,\%$ noted by Small [59]. DMPG shows a small increase in the tail volume when compared with the literature value, albeit at a higher temperature. However, this value agrees well with that found for DMPC which shares the same tail structure.

3.4.5 Solvent effect on the lipid head group volume

Tables 3.3, 3.4, 3.5, and 3.6 give the best-fit values for the head group volumes for each of phospholipids investigated. Ths three phospholipids with the PC head group are consistent, giving values of $\sim 330\,\text{Å}^3$, regardless of the tail group. This agrees well with the values found for the same head component in water, shown in Table 3.1. Interestingly, the head group volume determined for the PG containing phospholipid is similar to that for the PC head group, with a value of 329.96 ± 0.34 . The PG head group volume in water, from either DMPG using differential vibrating tube densimetry [72] or POPG using molecular dynamics simulations [73], is noticeably smaller. This indicates that there may be some effect arising from the solvation of the PG component in the choline chloride:glycerol DES. However, this has only been shown for a single PG-lipid at the air-DES interface.

The major difference between the two head groups of the lipid is that the PG is present as a sodium salt, whereas the PC is zwitterionic. When in solution the anionic PG head is expected to associate with cations in solution, as it does in water [88] where sucj interactions

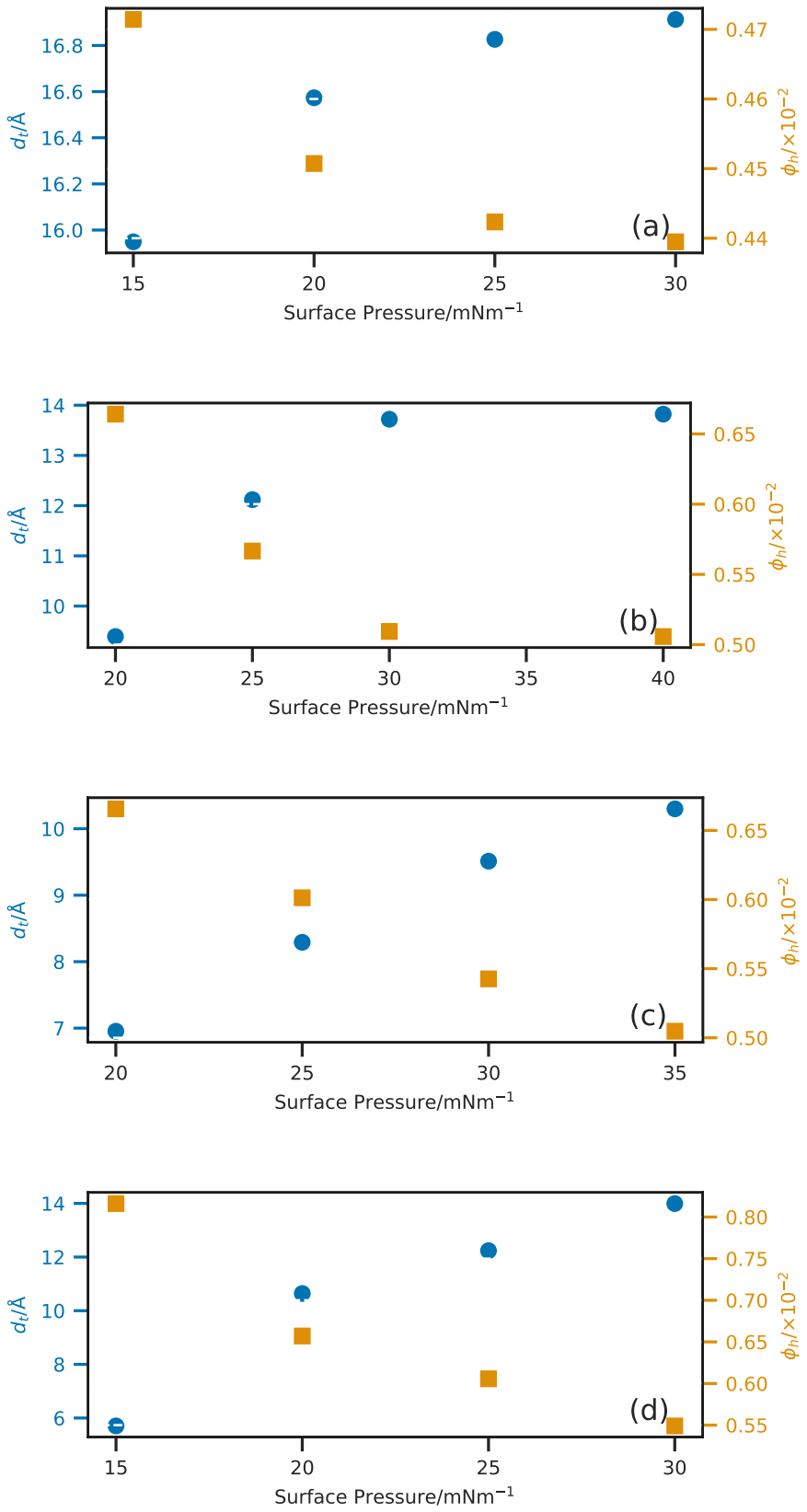


FIGURE 3.5: The variation of d_t (blue circles) and ϕ_h (orange squares) with surface pressures for each of the four phospholipids; (a) DPPC, (b) DMPC, (c) DLPC, and (d) DMPG.

depend on a variey of factors including the ionic strength. In the case of a DES, the environment is inherent ionic and therefore the interaction of an anionic lipid head may be more complex. As well as interacting with this sodium, the head is likely to interact with the choline cations, similar to behaviour reported previously for surfactact micelles [89]. The extent of interaction with each of the cations is unclear, but regardless it seems likely that the solvation of the PH head is improved in the DES relative to water. This better solvation would explain the apparent increase in the volume of the PG head since it would result in a swelling of this group through its strong interactions with the solvent. In the case of PC, the proximity of a local cation within the molecule results in the same folding of the head group seen in water because this interaction is less transient that the equivalent interactions with the solvent.

3.4.6 Analysis of neutron reflectometry

The ability to fit NR data in Figure 3.4 indicated that the values found for the head and tail groups are consistent between the pair of measurements for the same systems. It is clear, that again stable monolayers of the phospholipids are forming at the air-DES interface and that the volume determined by XRR measurements are robust enough to be used in the modelling of NR data. Furthermore, as shown in Table 3.7, the trends observed with increasing surface pressure in the XRR models, pertaining to a responsive increase in tail thickness and a decrease in solvent concentration in the head layer are consistent with that found with the NR analysis.

3.4.7 Utility of Markov chain Monte Carlo sampling

The use of MCMC sampling enabled the inverse uncertainties for each of the fitted parameters to be determined as a confidence interval from the PDFs shown in Figures 3.6, 3.7, 3.8, 3.9, and 3.10. These confidence intervals are useful for understanding the probability of the given value, however as discussed in Section 3.3 these intervals only represent the uncertainty in the experimental data and do not account for systematic uncertainty present in the measurement method.

The sums of the magnitudes of the Pearson correlation coefficients for each PC-containing phospholipids at each surface pressures are given in Figure 3.11. From this figure, it is clear that there is a relationship between the phospholipid and the correlation present in the reflectometry model. This is the effect of the reduction in the phospholipid length from DPPC to DLPC, and that a corresponding decrease is not observed for the interfacial roughness. Therefore, the boundary between phospholipid head and tail layers is less well defined, this can be observed by investigating the SLD profiles in Figure 3.3. Furthermore, the magnitude of the Pearson correlation between the head and tail thicknesses increases with increasing tail length; from -0.89 for DPPC, to -0.92 for DMPC, to -0.99 for DLPC, each at $30\,\mathrm{mN\,m^{-1}}$. Indicating that as the phospholipid tail length decreases the negative correlation between the head and tail layers increases to the point for DLPC where the two variables are almost completely correlated and the boundary between the tail is nearly nonexistent.

There is also a substantial positive correlation present for all of the datasets between the phospholipid head layer thickness and the volume fraction of solvent in the head layer. This correlation can be rationalised as a result of the SLD of the solvent and the head layer (which is much as $78\,\%$ solvent by volume) being similar, and therefore the boundary between the head layer and the solvent is also poorly defined. A significant correlation such

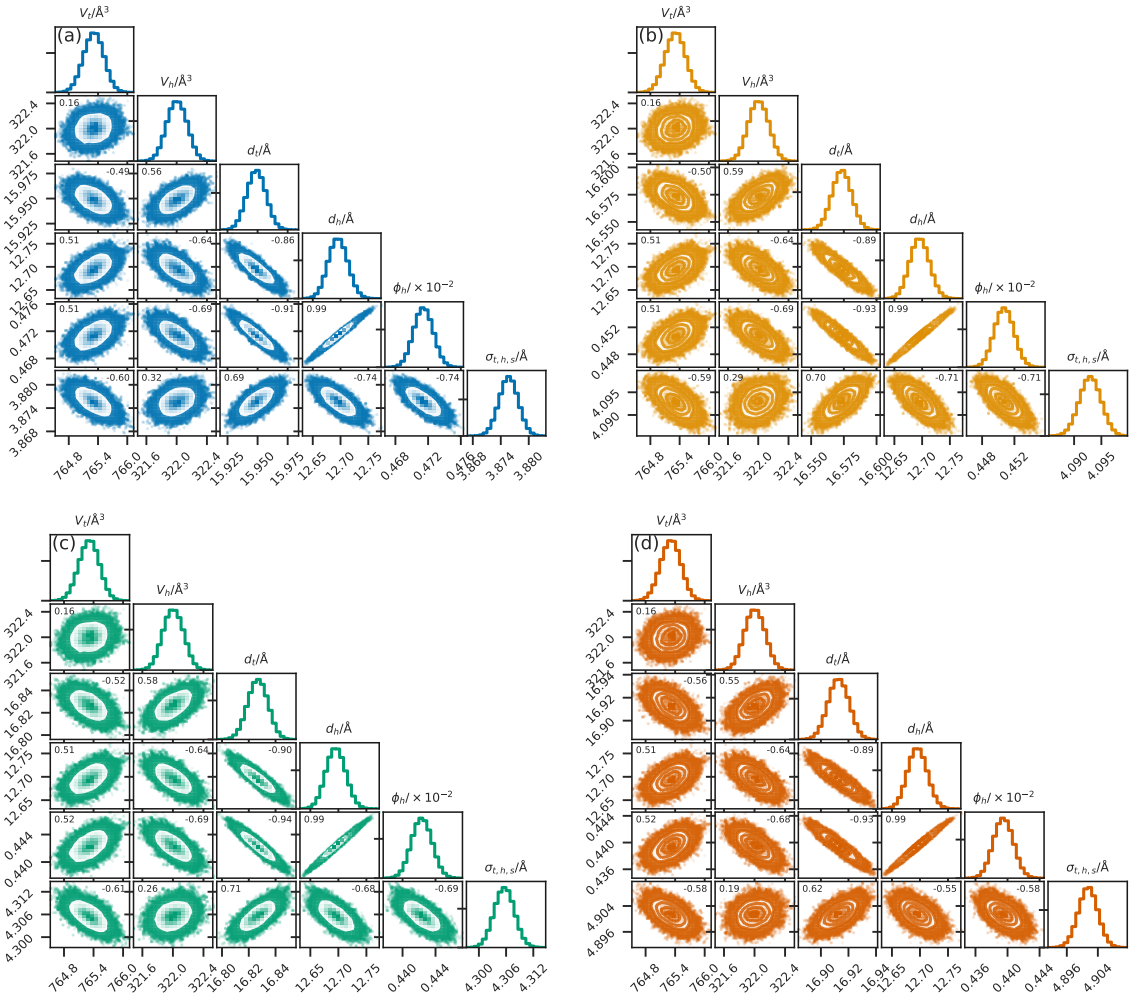


FIGURE 3.6: The probability distribution functions from the chemically-consistent modelling of DPPC; (a) at $15\,\mathrm{mN\,m^{-1}}$, (b) at $20\,\mathrm{mN\,m^{-1}}$, (c) at $25\,\mathrm{mN\,m^{-1}}$, (d) at $30\,\mathrm{mN\,m^{-1}}$. The Pearson correlation coefficient for each pair of parameters is given in the top corner of each two-dimensional PDF.

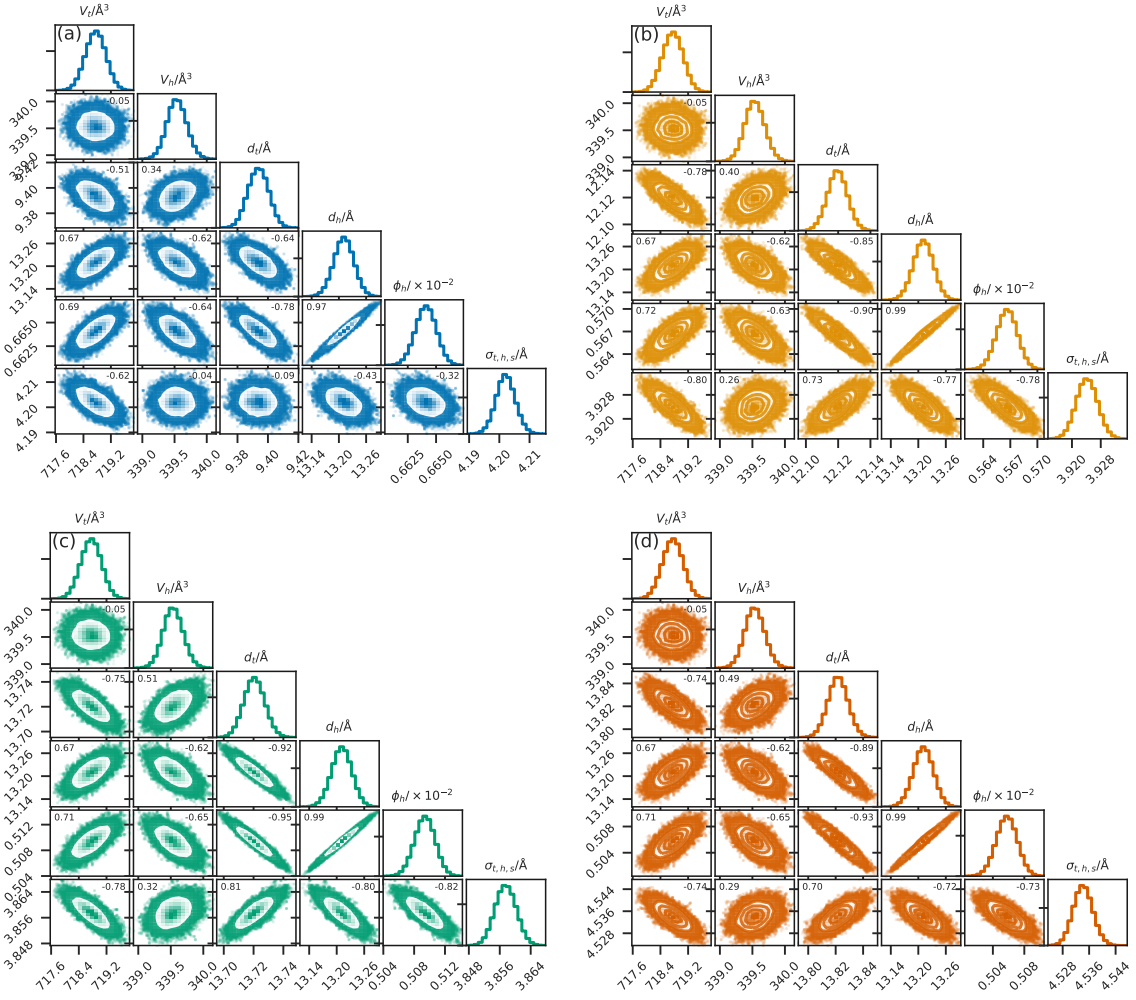


FIGURE 3.7: The probability distribution functions from the chemically-consistent modelling of DMPC; (a) at 20 mN m^{-1}, (b) at 25 mN m^{-1}, (c) at 30 mN m^{-1}, (d) at 40 mN m^{-1}. The Pearson correlation coefficient for each pair of parameters is given in the top corner of each two-dimensional PDF.

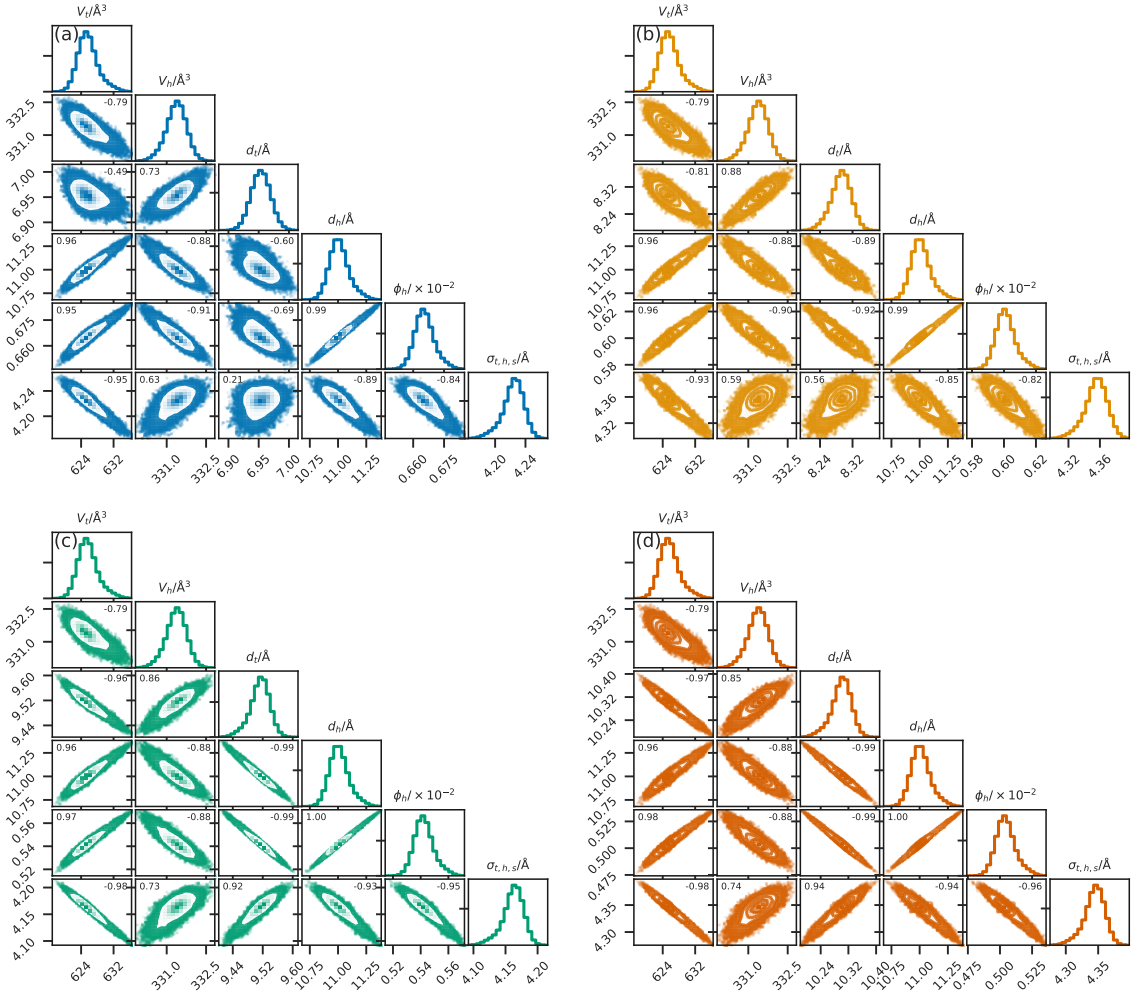


FIGURE 3.8: The probability distribution functions from the chemically-consistent modelling of DLPC; (a) at 20 mN m^{-1}, (b) at 25 mN m^{-1}, (c) at 30 mN m^{-1}, (d) at 35 mN m^{-1}. The Pearson correlation coefficient for each pair of parameters is given in the top corner of each two-dimensional PDF.

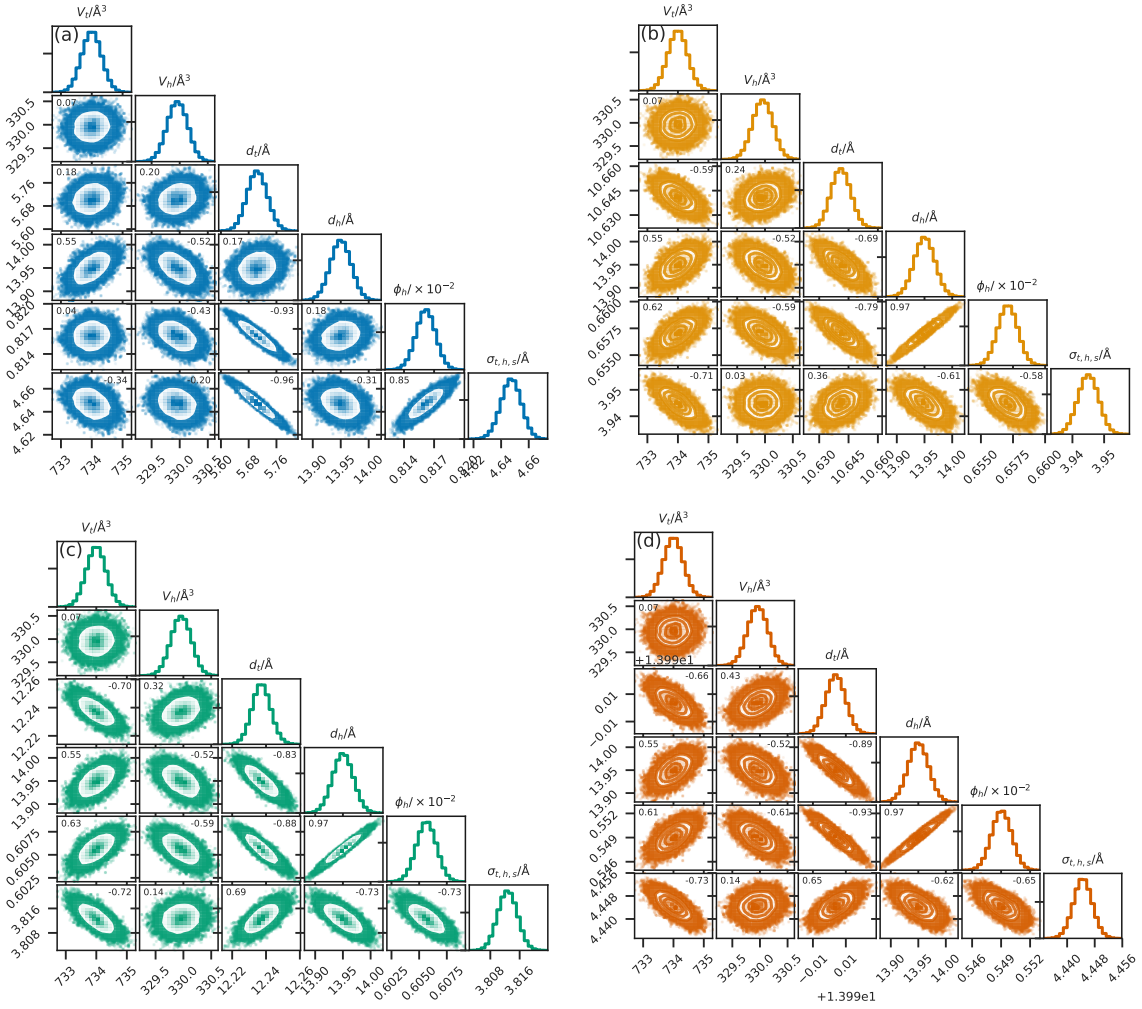


FIGURE 3.9: The probability distribution functions from the chemically-consistent modelling of DMPG; (a) at $15\,\mathrm{mN\,m^{-1}}$, (b) at $20\,\mathrm{mN\,m^{-1}}$, (c) at $25\,\mathrm{mN\,m^{-1}}$, (d) at $30\,\mathrm{mN\,m^{-1}}$. The Pearson correlation coefficient for each pair of parameters is given in the top corner of each two-dimensional PDF.

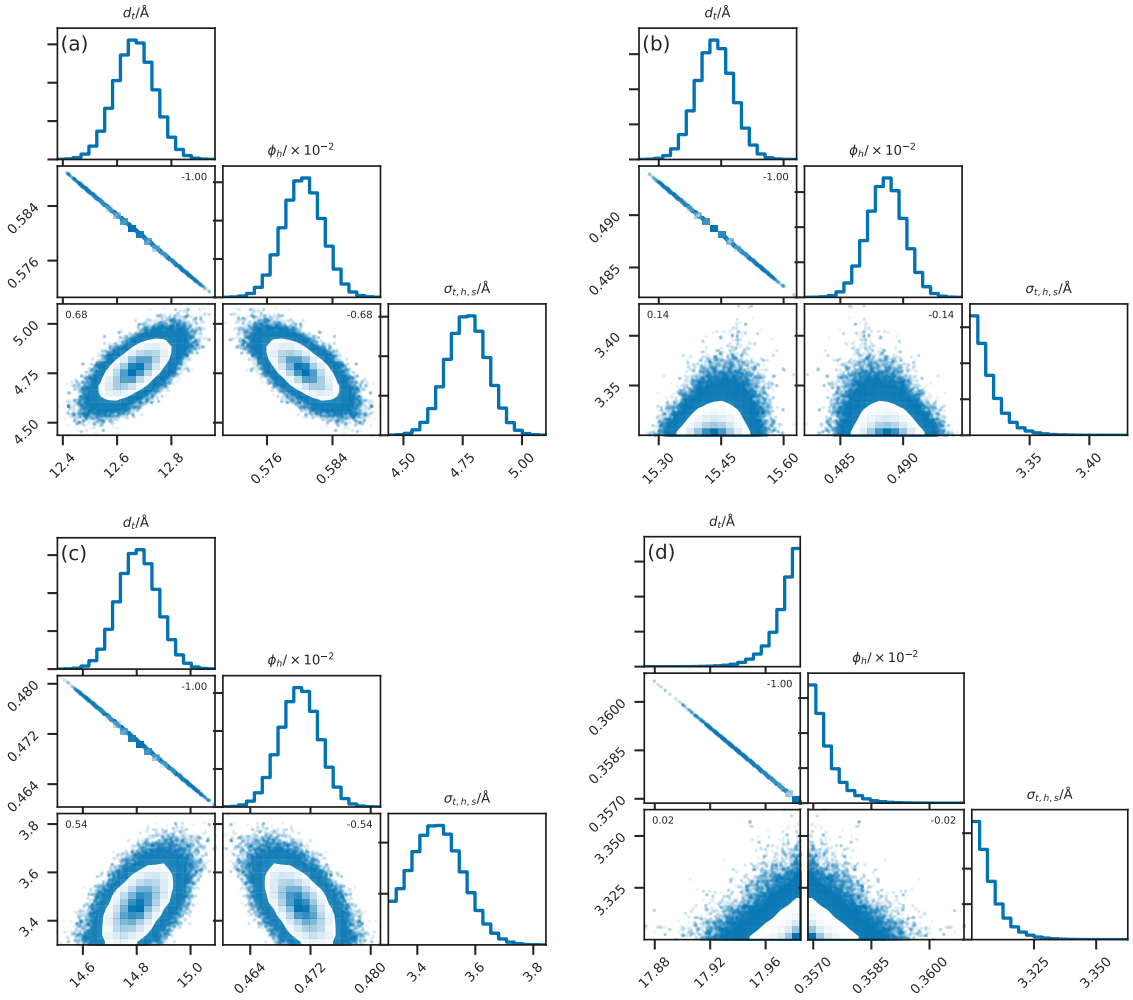


FIGURE 3.10: The probability distribution functions from the chemically-consistent modelling of the NR data; (a) DPPC at $15\,\mathrm{mN\,m^{-1}}$, (b) DPPC at $20\,\mathrm{mN\,m^{-1}}$, (c) DMPC at $20\,\mathrm{mN\,m^{-1}}$, and (d) DMPC at $25\,\mathrm{mN\,m^{-1}}$. The Pearson correlation coefficient for each pair of parameters is given in the top corner of each two-dimensional PDF.

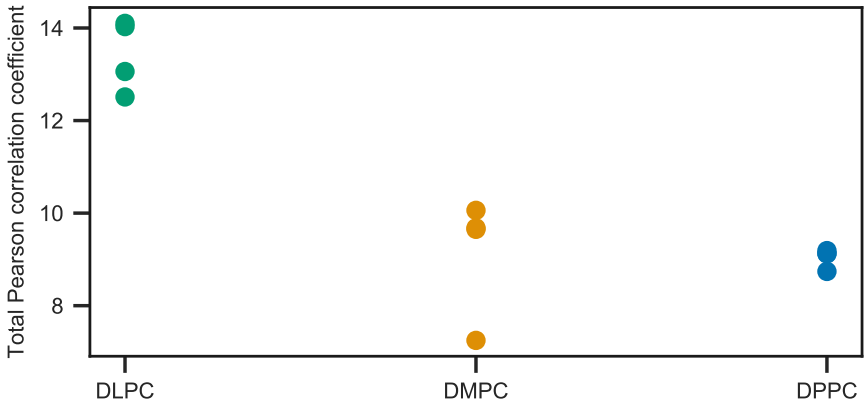


FIGURE 3.11: The sum of the magnitudes for each of the Pearson correlation coefficients for each of the PC-containing lipids.

as this is unavoidable, without considering the use of many neutron contrasts for both the phospholipid and the solvent, due to the solvophilic nature of the head groups.

3.5 Conclusions

Stable phosphocholine and phosphatidylglycerol lipid monolayers were observed and characterised on an ionic solvent surface. Until the emergence of ionic liquids and DES, only a limited number of molecular solvents exhibited the ability to promote self-assembly and only water and formamide among those had previously demonstrated the formation of phospholipid monolayers at the air-liquid interface.

For the first time, a physically and chemically-consistent reflectometry modelling approach was used to co-refine XRR measurements at different surface pressures. This enabled modelling without the need to constrain the head and tail group volumes, enabling these parameters to vary freely to account for any variation occurring due to the elevated surface pressures used or the presence of a non-aqueous solvent, compared to the commonly applied literature values. This allows a significantly different in the PG head group volume to be observed; having a larger volume than observed for the same system in water. This suggests that the transfer of lipids to a DES is not just a simple substitution of the subphase. In this specific case, we have proposed an explanation based on the dissociation of the PG head group ssalt and the subsequent interaction with the DES.

Finally, MCMC sampling was used to understand the inverse uncertainties present in the modelling parameters, enabling a confidence interval to be quoted alongside the most probable value. The use of MCMC sampling also allowed the quantification of the correlations between the parameters of the chemically-consistent model. This show the significant correlations between the head layer thickness and the volume fraction of solvent, and the head layer thickness and the tail layer thickness that becomes more prominent for short-tailed phospholipids. The quantification of these correlations gives us a better understanding of the uncertainties on the parameters and can be rationalised based on the chemistry of the monolayers.

3.6 References

[1] E. L. Smith, A. P. Abbott and K. S. Ryder, *Chem. Rev.*, 2014, **114**, 11060–11082.
[2] Y. Dai, J. van Spronsen, G.-J. Witkamp, R. Verpoorte and Y. H. Choi, *Anal. Chim. Acta*, 2013, **766**, 61–68.
[3] O. S. Hammond, D. T. Bowron and K. J. Edler, *Green Chem.*, 2016, **18**, 2736–2744.
[4] O. S. Hammond, D. T. Bowron, A. J. Jackson, T. Arnold, A. Sanchez-Fernandez, N. Tsapatsaris, V. Garcia Sakai and K. J. Edler, *J. Phys. Chem. B*, 2017, **121**, 7473–7483.
[5] C. F. Araujo, J. A. P. Coutinho, M. M. Nolasco, S. F. Parker, P. J. A. Ribeiro-Claro, S. Rudić, B. I. G. Soares and P. D. Vaz, *Phys. Chem. Chem. Phys.*, 2017, **19**, 17998–18009.
[6] A. Pandey, R. Rai, M. Pal and S. Pandey, *Phys. Chem. Chem. Phys.*, 2014, **16**, 1559–1568.
[7] S. Zahn, B. Kirchner and D. Mollenhauer, *ChemPhysChem*, 2016, **17**, 3354–3358.
[8] B. D. Ribeiro, C. Florindo, L. C. Iff, M. A. Z. Coelho and I. M. Marrucho, *ACS Sustain. Chem. Eng.*, 2015, **3**, 2469–2477.
[9] D. J. G. P. van Osch, L. F. Zubeir, A. van den Bruinhorst, M. A. A. Rocha and M. C. Kroon, *Green Chem.*, 2015, **17**, 4518–4521.

[10] A. Sanchez-Fernandez, T. Arnold, A. J. Jackson, S. L. Fussell, R. K. Heenan, R. A. Campbell and K. J. Edler, *Phys. Chem. Chem. Phys.*, 2016, **18**, 33240–33249.
[11] T. Arnold, A. J. Jackson, A. Sanchez-Fernandez, D. Magnone, A. E. Terry and K. J. Edler, *Langmuir*, 2015, **31**, 12894–12902.
[12] Y.-T. Hsieh and Y.-R. Liu, *Langmuir*, 2018, **34**, 10270–10275.
[13] M. K. Banjare, K. Behera, M. L. Satnami, S. Pandey and K. K. Ghosh, *RSC Adv.*, 2018, **8**, 7969–7979.
[14] S. J. Bryant, R. Atkin and G. G. Warr, *Soft Matter*, 2016, **12**, 1645–1648.
[15] S. J. Bryant, R. Atkin and G. G. Warr, *Langmuir*, 2017, **33**, 6878–6884.
[16] M. G. Gutiérrez, M. L. Ferrer, C. R. Mateo and F. del Monte, *Langmuir*, 2009, **25**, 5509–5515.
[17] L. Sapir, C. B. Stanley and D. Harries, *J. Phys. Chem. A*, 2016, **120**, 3253–3259.
[18] A. Sanchez-Fernandez, K. J. Edler, T. Arnold, D. Alba Venero and A. J. Jackson, *Phys. Chem. Chem. Phys.*, 2017, **19**, 8667–8670.
[19] H. Mohwald, *Annu. Rev. Phys. Chem.*, 1990, **41**, 441–476.
[20] S. Kewalramani, H. Hlaing, B. M. Ocko, I. Kuzmenko and M. Fukuto, *J. Phys. Chem. Lett.*, 2010, **1**, 489–495.
[21] F. Graner, S. Perez-Oyarzun, A. Saint-Jalmes, C. Flament and F. Gallet, *J. Phys. II France*, 1995, **5**, 313–322.
[22] S. P. Weinbach, K. Kjaer, J. Als-Nielsen, M. Lahav and L. Leiserowitz, *J. Phys. Chem.*, 1993, **97**, 5200–5203.
[23] O. M. Magnussen, B. M. Ocko, M. Deutsch, M. J. Regan, P. S. Pershan, D. Abernathy, G. Grübel and J.-F. Legrand, *Nature*, 1996, **384**, 250–252.
[24] H. Kraack, B. M. Ocko, P. S. Pershan, E. Sloutskin and M. Deutsch, *Science*, 2002, **298**, 1404–1407.
[25] M. R. Lovell and R. M. Richardson, *Curr. Opin. Colloid Interface Sci.*, 1999, **4**, 197–204.
[26] M. Geoghegan, R. A. L. Jones, D. S. Sivia, J. Penfold and A. S. Clough, *Phys. Rev. E*, 1996, **53**, 825–837.
[27] D. G. Bucknall, S. A. Butler, H. E. Hermes and J. S. Higgins, *Physica B*, 1997, **241-243**, 1071–1073.
[28] D. S. Sivia, W. I. F. David, K. S. Knight and S. F. Gull, *Physica D*, 1993, **66**, 234–242.
[29] D. S. Sivia and J. R. P. Webster, *Physica B*, 1998, **248**, 327–337.
[30] D. S. Sivia, W. A. Hamilton and G. S. Smith, *Physica B*, 1991, **173**, 121–138.
[31] X.-L. Zhou and S.-H. Chen, *Physical Review E*, 1993, **47**, 3174–3190.
[32] X.-L. Zhou and S.-H. Chen, *Phys. Rep.*, 1995, **257**, 223–348.
[33] K. Kunz, J. Reiter, A. Goetzelmann and M. Stamm, *Macromolecules*, 1993, **26**, 4316–4323.
[34] V.-O. de Haan and G. G. Drijkoningen, *Physica B*, 1994, **198**, 24–26.
[35] A. D. Dane, A. Veldhuis, D. K. G. de Boer, A. J. G. Leenaers and L. M. C. Buydens, *Physica B*, 1998, **253**, 254–268.
[36] A. Ulyanenkov, K. Omote and J. Harada, *Physica B*, 2000, **283**, 237–241.
[37] A. Ulyanenkov and S. Sobolewski, *J. Phys. D*, 2005, **38**, A235–A238.
[38] E. Politsch and G. Cevc, *J. Appl. Crystallogr.*, 2002, **35**, 347–355.
[39] M. Wormington, C. Panaccione, K. M. Matney and D. K. Bowen, *Philos. Trans. R. Soc. London Ser. A*, 1999, **357**, 2827–2848.
[40] M. Björck, *J. Appl. Crystallogr.*, 2011, **44**, 1198–1204.
[41] M. Björck and G. Andersson, *J. Appl. Crystallogr.*, 2007, **40**, 1174–1178.
[42] A. R. J. Nelson, *J. Appl. Crystallogr.*, 2006, **39**, 273–276.
[43] A. R. J. Nelson and S. W. Prescott, *J. Appl. Crystallogr.*, 2019, **52**, 193–200.
[44] F. Ott, *SimulReflec*, `http://www-llb.cea.fr/prism/programs/simulreflec/simulreflec.html`, (accessed 2019-03-04).

[45] P. A. Kienzle, M. Douchet, D. J. McGilivray, K. V. O'Donovan, N. K. Berk and C. F. Majkrzak, *NCNR Reflectometry Software*, `http://www.ncnr.nist.gov/reflpak`, (accessed 2018-03-04).
[46] D. L. Gil and D. Windover, *J. Phys. D*, 2012, **45**, 235301.
[47] D. P. Hoogerheide, S. Y. Noskov, A. J. Kuszak, S. K. Buchanan, T. K. Rostovtseva and H. Nanda, *Acta Crystallogr. D*, 2018, **74**, 1219–1232.
[48] J. E. Owejan, J. P. Owejan, S. C. DeCaluwe and J. A. Dura, *Chem. Mater.*, 2012, **24**, 2133–2140.
[49] F. Heinrich, H. Nanda, H. Z. Goh, C. Bachert, M. Lösche and A. D. Linstedt, *J. Biol. Chem.*, 2014, **289**, 9683–9691.
[50] R. Storn and K. Price, *J. Global Optim.*, 1997, **11**, 341–359.
[51] E. Jones, T. Oliphant, P. Peterson and others, *SciPy: Open Source Scientific Tools for Python*, `http://www.scipy.org`, (accessed 2019-03-04).
[52] D. Foreman-Mackey, D. W. Hogg, D. Lang and J. Goodman, *Publ. Astron. Soc. PAc.*, 2013, **125**, 306–312.
[53] R. A. Campbell, Y. Saaka, Y. Shao, Y. Gerelli, R. Cubitt, E. Nazaruk, D. Matyszewska and M. J. Lawrence, *J. Colloid Interface Sci.*, 2018, **531**, 98–108.
[54] K. Wojciechowski, M. Orczyk, T. Gutberlet, G. Brezesinski, T. Geue and P. Fontaine, *Langmuir*, 2016, **32**, 9064–9073.
[55] K. Wojciechowski, M. Orczyk, T. Gutberlet and T. Geue, *BBA - Biomembranes*, 2016, **1858**, 363–373.
[56] F. Foglia, G. Fragneto, L. A. Clifton, M. J. Lawrence and D. J. Barlow, *Langmuir*, 2014, **30**, 9147–9156.
[57] G. Bello, A. Bodin, M. J. Lawrence, D. Barlow, A. J. Mason, R. D. Barker and R. D. Harvey, *BBA - Biomembranes*, 2016, **1858**, 197–209.
[58] C. W. McConlogue and T. K. Vanderlick, *Langmuir*, 1997, **13**, 7158–7164.
[59] D. Small, *J. Lipids Res.*, 1984, **25**, 1490–1500.
[60] C. M. Hollinshead, R. D. Harvey, D. J. Barlow, J. R. P. Webster, A. V. Hughes, A. Weston and M. J. Lawrence, *Langmuir*, 2009, **25**, 4070–4077.
[61] T. Arnold, C. Nicklin, J. Rawle, J. Sutter, T. Bates, B. Nutter, G. McIntyre and M. Burt, *J. Synchotron Radiat.*, 2012, **19**, 408–416.
[62] R. A. Campbell, H. P. Wacklin, I. Sutton, R. Cubitt and G. Fragneto, *Eur. Phys. J. Plus*, 2011, **126**, 107.
[63] T. M. Bayerl, R. K. Thomas, J. Penfold, A. Rennie and E. Sackmann, *Biophys. J.*, 1990, **57**, 1095–1098.
[64] S. J. Johnson, T. M. Bayerl, D. C. McDermott, G. W. Adam, A. R. Rennie, R. K. Thomas and E. Sackmann, *Biophys. J.*, 1991, **59**, 289–294.
[65] L. A. Clifton, M. Sanders, C. Kinane, T. Arnold, K. J. Edler, C. Neylon, R. J. Green and R. F. Frazier, *Phys. Chem. Chem. Phys.*, 2012, **14**, 13569.
[66] C. A. Helm, H. Möhwald, K. Kjær and J. Als-Nielsen, *EPL*, 1987, **4**, 697–703.
[67] J. Daillant, L. Bosio and J. J. Benattar, *EPL*, 1990, **12**, 715–720.
[68] R. S. Armen, O. D. Uitto and S. E. Feller, *Biophys. J.*, 1998, **75**, 734–744.
[69] W.-J. Sun, R. M. Suter, M. A. Knewtson, C. R. Worthington, S. Tristram-Nagle, R. Zhang and J. F. Nagle, *Phys. Rev. E*, 1994, **49**, 4665–4676.
[70] N. Kučerka, M. A. Kiselev and P. Balgavý, *Eur. Biophys. J.*, 2004, **33**, 328–334.
[71] P. Balgavý, N. Kučerka, V. I. Gordeliy and V. G. Cherezov, *Acta. Phys. Slovaca*, 2001, **51**, 53–68.
[72] J. Pan, F. A. Heberle, S. Tristram-Nagle, M. Szymanski, M. Koepfinger, J. Katsaras and N. Kučerka, *BBA - Biomembranes*, 2012, **1818**, 2135–2148.
[73] N. Kučerka, B. W. Holland, C. G. Gray, B. Tomberli and J. Katsaras, *J. Phys. Chem. B*, 2012, **116**, 232–239.

[74] A. Sanchez-Fernandez, G. L. Moody, L. C. Murfin, T. Arnold, A. J. Jackson, S. M. King, S. E. Lewis and K. J. Edler, *Soft Matter*, 2018, **14**, 5525–5536.
[75] D. Marsh, *Chem. Phy. Lipids*, 2010, **163**, 667–677.
[76] A. R. J. Nelson, S. W. Prescott, I. Gresham and A. R. McCluskey, *Refnx v0.1.2*, `http://doi.org/10.5281/zenodo.2552023`, 2019.
[77] F. Abelès, *Ann. Phys.*, 1948, **12**, 504–520.
[78] L. G. Parratt, *Phys. Rev.*, 1954, **95**, 359–369.
[79] A. R. McCluskey, A. Sanchez-Fernandez, K. J. Edler, S. C. Parker, A. J. Jackson, R. A. Campbell and T. Arnold, *Lipids_at_airdes (Version 1.0)*, `http://doi.org/10.5281/zenodo.2577796`, 2019.
[80] C. Tanford, *The Hydrophobic Effect: Formation of Micelles and Biological Membranes*, John Wiley & Sons, New York, 2nd edn., 1980.
[81] L. Braun, M. Uhlig, R. von Klitzing and R. A. Campbell, *Adv. Colloid Interface Sci.*, 2017, **247**, 130–148.
[82] E. B. Watkins, C. E. Miller, D. J. Mulder, T. L. Kuhl and J. Majewski, *Phys. Rev. Lett.*, 2009, **102**, 238101–238104.
[83] Y. Gerelli, *J. Appl. Crystallogr.*, 2016, **49**, 712–712.
[84] J. Goodman and J. Weare, *Comm. App. Math. Comp. Sci.*, 2010, **5**, 65–80.
[85] S. S. Shapiro and M. B. Wilk, *Biometrika*, 1965, **52**, 591–611.
[86] K. Pearson, *Proc. Royal Soc. Lond.*, 1895, **58**, 240–242.
[87] D. Vaknin, K. Kjaer, J. Als-Nielsen and M. Lösche, *Biophys. J.*, 1991, **59**, 1325–1332.
[88] D. Grigoriev, R. Krustev, R. Miller and U. Pison, *J. Phys. Chem. B*, 1999, **103**, 1013–1018.
[89] A. Sanchez-Fernandez, O. S. Hammond, K. J. Edler, T. Arnold, J. Doutch, R. M. Dalgliesh, P. Li, K. Ma and A. J. Jackson, *Phys. Chem. Chem. Phys.*, 2018, **20**, 13952–13961.

4 Applying atomistic and coarse-grained simulation to reflectometry analysis

Abstract

The use of molecular simulation to aid in the analysis of neutron reflectometry measurements is commonplace. However, reflectometry is a tool to probe large-scale structures, and therefore the use of all-atom simulation may be irrelevant. This work presents the first direct comparison between the reflectometry profiles obtained from different all-atom and coarse-grained molecular dynamics simulations and the reflectometry profiles from a chemically-consistent layer modelling method. We find that systematic limitations reduce the efficacy of the MARTINI potential model, while the Berger united-atom and Slipids all-atom potential models agree similarly well with the experimental data. The chemically-consistent layer model gives the best agreement, however, the higher resolution simulation-dependent methods produce an agreement that is comparable.

Context

This chapter builds on the previous chapter, by using the traditional, highly coarse-grained, chemically-consistent layer model as a point of comparison with classical molecular simulations of a variety of simulation grain-size, from the coarse-grained MARTINI potential model, to the all-atom Slipids. Therefore, the chapter applies significantly more constraints on the system, however, the constraints are built on substantial underlying chemistry, given there grounding in the potential models. It is hoped that this work will provide as an advisory document to those interested in applying classical simulation to the analysis of their neutron reflectometry experiments. Furthermore, the simulations are used to advise on ways that the chemically-consistent models may be improved in the future. It is noted again that the focus of this chapter is the methodological developments, rather than the particular chemical system to which they are applied.

4.1 Introduction

The use of a traditional, layer model, approach as outlined in Chapter 3, may not be sufficient for the study of complex systems such as biomimetic bacterial membranes [1] and polymeric energy materials [2]. A commonly suggested method for the multi-modal analysis of reflectometry measurements is the use of molecular dynamics (MD) simulation. MD-driven multi-modal analysis has been applied previously, either by the calculation of the scattering length density (SLD) profile from the simulation of by the full determination fo the reflectometry profile. In the former case, the calculated SLD profile may be compared with the SLD profile determined from the use of a traditional analysis method. Bobone *et al.* used such a method to study the antimicrobial peptide trichogin GA-IV within a supported lipid bilayer [3]. A four-layer model consisting of the hydrated SiO_2 layer, an inner lipid head-region, a lipid tail-region, and an outer lipid head region was used in the Abelès matrix formalism. The SLD profile from the MD simulations agreed well with that fitted to the reflectometry data from the layer model.

The reflectometry profile was determined explicitly from the classical simulation in the works of Miller *et al.* and Anderson and Wilson [4, 5]. In these works, an amphiphilic polymer at the oil-water interface was simulated by Monte Carlo and MD respectively, and the neutron reflectometry profile was found by splitting the simulation cell into a series of small layers and treating these layers with the Abelès formalism. There was good agreement between the experimental and calculated reflectometry, for low interfacial coverages of the polymer. Another work that has made a direct comparison between the atomistic simulation-derived reflectometry data and those measured experimentally includes that of Darré *et al.* [6]. In this work, NeutronRefTools was developed to produce the neutron reflectometry profile from an MD simulation. The particular system studied was a supported 1,2-dimyristoyl-2*n*-glycero-3-phosphocholine (DMPC) lipid bilayer, with good agreement found between the simulation-derived profile and the associated experimental measurements. However, the nature of the support required that a correction for the head group hydration be imposed to achieve this agreement.

Koutsioubas used the MARTINI coarse-grained representation of a 1,2-dipalmitoyl-*sn*-glycero-3-phosphocholine (DPPC) lipid bilayer to compare with experimental reflectometry [7]. This work shows that the parameterisation of the MARTINI water bead was extremely important in the reproduction of the reflectometry data, as the non-polarisable water bead would freeze into crystalline sheets resulting in artefacts in the reflectometry profiles calculated. The work of Hughes *et al.* studied again a DPPC lipid bilayer system [8], albeit an all-atom representation, that was compared with a supported DPPC lipid bilayer system measured with polarised neutron reflectometry. The SLD profile found from MD simulation was varied to better fit the experimental measurement, resulting in good agreement. Additionally, the ability to vary the SLD profile was used to remove artefact that arose when the MD simulations were merged with an Abelès layer model. This was done to account for regions present in the experiment that were not modelled explicitly.

In all the examples discussed so far, there is no direct comparison between the reflectometry profile determined from simulation and that from the application of a traditional modelling approach. Indeed, the only example, to the authors' knowledge, where a direct comparison was drawn is the work of Dabkowska *et al.* [9]. This work compares the reflectometry profile from a DPPC monolayer at the air-water interface containing dimethyl sulfoxide molecules with a similar molecular dynamics simulation using the CHARMM potential model. The use of multimodal analysis allowed the determination of the position and orientation of DMSO molecules at a particular region within the monolayer.

The previously mentioned work of Koutsioubas involved the use of the MARTINI coarse-grained force field to simulations the DPPC bilayer system [7]. The use of atomistic simulation for soft matter systems, such as a lipid bilayer, is undesirable as this requires a huge number of atoms to be simulated, due to the large lengths scales involved. The purpose of simulation coarse-graining is to reduce the number of particles over which the forces must be integrated, additionally by removing the higher frequency bond vibrations, the simulation timestep can also be increased [10]. Together, these two factors enable an increase in both simulation size and length. The use of the MARTINI 4-to-1 coarse-grained and the Berger united atom, where hydrogen atoms are integrated into the heavier atoms to which they are bound, potential models are particularly pertinent for the application to lipid simulations as both were developed with this specific application in mind [11, 12].

The MARTINI potential model involves integrating the interactions of every four heavy atoms, e.g. larger than hydrogen, into beads of different chemical nature. This potential model attempts to simplify the interactions of lipid and protein molecules significantly by allowing for only eighteen particle types, defined by their polarity, charge, and hydrogen-bond acceptor/donor character, which are discussed in detail in the work of Marrink *et al.* [11]. This coarse-grained potential model was initially developed for the simulation of a lipid bilayer, and proteins held within and therefore is parameterised well under these conditions. It has successfully been used to simulate a wide range of systems, such as DNA nucleotides [13], the micellisation of zwitterionic and nonionic surfactants [14], and the self-assembly of ionic surfactants [15].

Increasing the simulation resolution gives an united-atom potential mode, where all of the hydrogen atoms are integrated into the heavier atoms to which they are bound. One of the most popular united-atom potential models for lipid simulations is that developed by Berger *et al.* [12]. The Berger parameters were optimised to reproduce lipid density and area per lipid, the latter of which is often an important parameter for the understanding of reflectometry profiles. Since it's inception, this potential model has proven one of the most commonly used and resilient sets of lipid parameters, with the original paper being cited 1500 times at the time of writing. Applications of this potential model have mostly been focussed on the simulation of membrane-bound proteins in a lipid bilayer [16, 17].

The Slipid (Stockholm Lipids) potential model was developed in 2012 by Jämbeck and Lyubartsev [18], where the potential model was again designed to reproduce the structure of a lipid bilayer. The authors optimised the average area per lipid, the thermal expansivity, and contractivity, among other structural and thermodynamic parameters. This included comparing the X-ray reflectometry profiles of the lipid bilayers with those measured experimentally. In later work, additional parameters were optimised to agree well with experimental values [19, 20]. Similar to the application of the Berger potential model, the Slipid potential model has been applied to the study of membrane-protein bound systems, such as the modulation of ion transfer [21]. However, it has also been used to the study of water diffusion within lipid membranes [22].

This chapter present the comparison of three MD simulations of different potential models, with different degree of coarse-graining; namely the Slipid all-atom [18], Berger united-atom [12], and MARTINI coarse-grained potential models [11]. This comparison offers a fundamental insight into the simulation resolution that is necessary to reproduce experimental neutron reflectometry measurements. Furthermore, we suggest some adjustments that may be made to the traditional, layer models that are commonly used to analyse these measurements.

TABLE 4.1: The different contrasts of phospholipid and water investigated. ACMW is air-contrast matched water, where D_2O and H_2O are mixed so as to give water with a SLD of zero.

Shorthand	Phospholipid contrast	Water contrast
h-D_2O	h-DSPC	D_2O
d_{13}-D_2O	d_{13}-DSPC	D_2O
d_{13}-ACMW	d_{13}-DSPC	ACMW
d_{70}-D_2O	d_{70}-DSPC	D_2O
d_{70}-ACMW	d_{70}-DSPC	ACMW
d_{83}-D_2O	d_{83}-DSPC	D_2O
d_{83}-ACMW	d_{83}-DSPC	ACMW

4.2 Methods

4.2.1 Neutron reflectometry measurements

The neutron reflectometry measurements analysed in this chapter were published previously by Hollinshead *et al.* [23], and full details of the experimental methods used can be found in that publication. The measurements conducted involved seven contrasts fo 1,2-distearoyl-*sn*-phosphatidylcholine (DSPC) at the air-water interface (Table 4.1), at four different surface pressures; 20 to $50\,\mathrm{mN\,m^{-1}}$.

4.2.2 Molecular dynamics simulations

The DSPC monolayer simulations were made up of lipid molecules modelled with three potential models, each with a different degree of coarse-graining. The Slipids potential model is an all-atom representation of the lipids molecules [18], which was used alongside the single point charge (SPC) water model [24], with a timestep of $0.5\,\mathrm{fs}$. The Berger potential model is obtained by the integration of the hydrogen atoms into the heavy atoms to which they are bound, producing a united-atom potential model [12]; again the SPC water molecules were used. The removal of the high-frequency C−H bonds allows the timestep to be increased to $1\,\mathrm{fs}$. Finally, the lowest resolution potential model used was the MARTINI [11] alongside the polarisable MARTINI water model [25], to avoid the freezing issues observed previously [7]. The MARTINI 4-to-1 heavy atom beading allows for the use of a $20\,\mathrm{fs}$ timestep. All simulations were conducted with temperature coupling to a heat bath at $300\,\mathrm{K}$ and a leapfrog integrator, and run using GROMACS 5.0.5 [26–29] on 32 cores of the STFC Scientific Computing resource SCARF. The simulations were of monolayers, therefore the Ewald 3DC correction was applied to allow for the use of x/y-only periodic boundary condition [30]. A close-packed "wall" of non-interacting dummy atoms was placed at each side of the simulation cell in the z-direction, to ensure that the atoms could not leave the simulation cell.

The starting simulation structure was generated using the molecular packing software Packmol [31]. This was used to produce a monolayer of 100 DSPC molecules, with the head group oriented to the bottom of the simulation cell. A $6\,\text{Å}$ layer of water was then added such that it overlapped the head groups, this was achieved with the `solvate` functionality in GROMACS 5.0.5. Examples, of the dry and wet monolayer for the Berger potential model, can be seen in Figure 4.1.

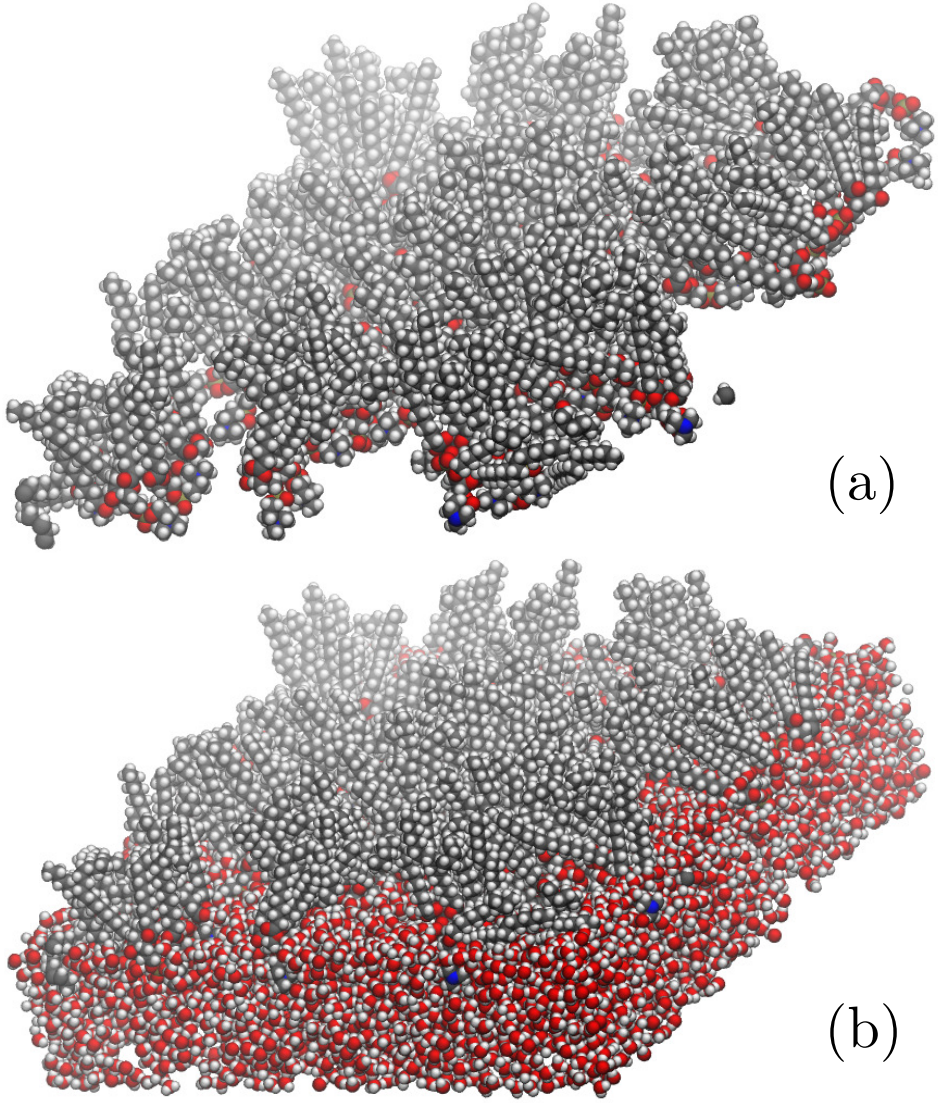


FIGURE 4.1: The DSPC monolayer; (a) without water layer and (b) with water layer, visuallised using VMD [32].

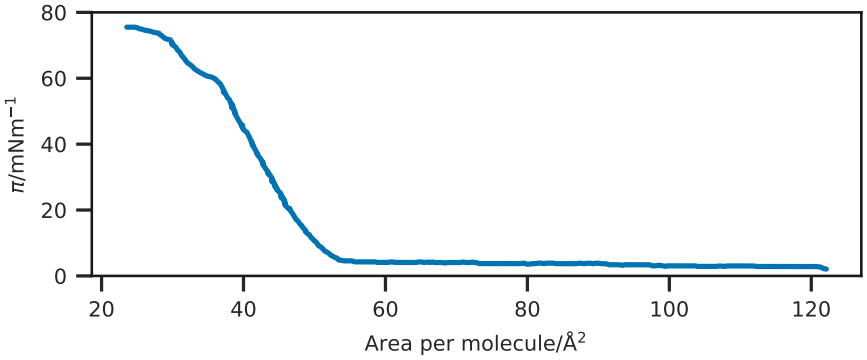


FIGURE 4.2: The experimental surface pressure isotherm for DSPC, taken from the work of Kubo *et al.* [33].

TABLE 4.2: The areas per molecule (APMs) associated with particular surface pressures and the size of the *x*- and *y*-cell dimension for a simulation of 100 phospholipid molecules.

$\pi/\mathrm{mN\,m^{-1}}$	APM/Å^2	*xy*-cell length/Å
20	47.9	69.1
30	46.4	68.1
40	45.0	67.1
50	44.6	66.0

A general protocol was then used to relax the system at the desired surface coverage, reproducing the effects of a Langmuir trough *in-silico*. This involved subjecting the system to a semi-isotropic barostat, with a compressibility of $4.5 \times 10^{-5}\,\mathrm{bar}^{-1}$ of the Slipids and Berger simulations and $3.0 \times 10^{-4}\,\mathrm{bar}^{-1}$ for the MARTINI simulations. The pressure is in the z-dimension was kept constant at $1\,\mathrm{bar}$, while it was increased in the *x*- and *y*-dimensions isotropically. When the *xy*-area is reached that is associated with the area per molecule (APM) for each surface pressure, described by the experimental surface pressure-isotherm (Figure 4.2), given in Table4.2, the coordinates were saved and used as the starting structure for the equilibration simulation. This involved continuing the use of the semi-isotropic barostat, with the *xy*-area of the box fixed, allowing the system to relax at a pressure of $1\,\mathrm{bar}$ in the z-dimension. The equilibration period was $1\,\mathrm{ns}$, following which the $50\,\mathrm{ns}$ NVT ensemble production simulations were run, on which all analyses were conducted.

4.3 Data analysis

4.3.1 Traditional layer-model analysis

In order to provide a point of comparison for the simulation-derived methods, the chemically-consistent reflectometry model developed in Chapter 3 was used for the analysis of the experimental data. The only modification that was made to the methodology described in Chapter 3 was that the volume of the phospholipid tail group, V_t was constrained based on the APM (taken from the surface pressure-isotherm data),

$$V_t = d_t \mathrm{APM}, \tag{4.1}$$

where d_t is the tail layer thickness. The result of this constraint is that both the monolayer model and the simulation-derived models were constrained equally by the measured surface coverage.

4.3.2 Simulation-dervied analysis

A custom-class, `md_simulation`, was developed for `refnx` [34, 35] at enabled the determination of a reflectometry profile from simulation, using a similar method to that employ in previous work, such as Dabkowska *et al.* [9]. The Abelès layer model formalism is applied to layers, the SLD of which is drawn directly from the simulation, and the thickness of which is defined. The layer thickness used was $1\,$Å for the Slipid and Berger potential model simulations, with an interfacial roughness between these layers of $0\,$Å. For the MARTINI potential model, a layer thickness of $4\,$Å was used, with an interfacial roughness of $0.2\,$Å. Each of the $50\,\mathrm{ns}$ production simulations were analysed each $0.1\,\mathrm{ns}$, and the SLD profiles were determined by summing the scattering lengths, b_j, for each of the atoms in a given layer.

$$\mathrm{SLD}_n = \frac{\sum_j b_j}{V_n}, \tag{4.2}$$

where, V_n is the volume of the layer n, obtains from the simulation cell parameters in the plane of the interface and the defined layer thickness. A uniform background was assigned based on the intensity at the highest q-vector and scale factor were then determined using `refnx` to offer the best agreement between the calculated reflectometry profile and that measured experimentally.

4.3.3 Comparison between monolayer model and simulation-derived analysis

In order to assess the agreement between the model from each method, the following goodness-of-fit metric was used, following the transformation of the data into Rq^4 space,

$$\chi^2 = \sum_{i=1}^{N_{\mathrm{data}}} \frac{[R_{\mathrm{exp}}(q_i) - R_{\mathrm{sim}}(q_i)]^2}{[\delta R_{\mathrm{exp}}(q_i)]^2}, \tag{4.3}$$

where, q_i is a given q-vector, $R_{\mathrm{exp}}(q_i)$ is the experimental reflected intensity, $R_{\mathrm{sim}}(q_i)$ is the simulation-derived/traditionally-developed reflected intensity, and $\delta R_{\mathrm{exp}}(q_i)$ is the resolution function of the experimental data.

Parametric outcomes from the different analysis methods were also compared, such as the number of water molecules per head group, wph. This was obtained from the monolayer model by considering the solvent fraction in the head-layer, ϕ_h, the volume of the head group, V_h, and taking the volume of a single water molecule to be $29.9\,\mathrm{Å}^3$ (from the density of water as $997\,\mathrm{kg\,m^{-3}}$),

$$\mathrm{wph} = \frac{\phi_h V_h}{29.9 - 29.9\phi_h}. \tag{4.4}$$

In MD simulations, the number density for each component in the system are obtained directly from the trajectory. The ratio of the lipid heads and the water then give the wph. The range for this was taken as between the $20\,\%$ and $80\,\%$ quantiles of the lipid head layers. Another parameter that was considered from the MD simulations was the length of the carbon tail in the molecules. This was determined as the length of a vector from the first atom, or bead, of the tail group to the last atom. This was calculated for each carbon tail

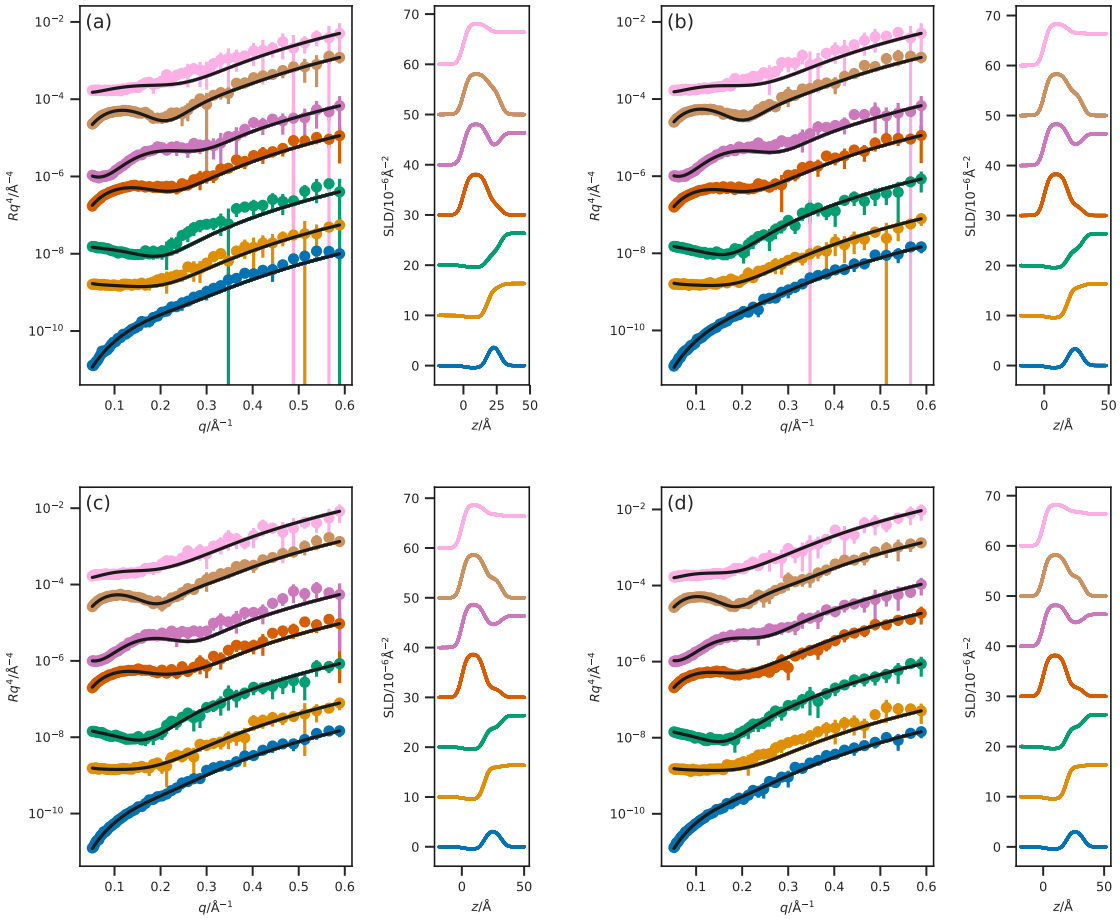


FIGURE 4.3: The NR profiles (left) and SLD profiles (right) determined from the chemically-consistent model for DSPC; (a) at $20\,\mathrm{mN\,m^{-1}}$, (b) at $30\,\mathrm{mN\,m^{-1}}$, (c) at $40\,\mathrm{mN\,m^{-1}}$, and (d) at $50\,\mathrm{mN\,m^{-1}}$. From top-to-bottom the contrasts are as follows; $d_8$3-D_2O, $d_8$3-ACMW, $d_7$0-D_2O, $d_7$0-ACMW, h-D_2O, $d_1$3-D_2O, $d_1$3-ACMW. The different contrast NR profiles have been offset in the y-axis by an order of magnitude and the SLD profiles offset in the y-axis by $10 \times 10^{-6}\,\mathrm{Å}^{-2}$, for clarity.

in each molecule, across all of the timesteps analysed, and are quoted as the median values with a $95\,\%$ quantile of the underlying sample.

4.4 Results & Discussion

Figures 4.3, and 4.6 compare the reflectometry and SLD profiles from each of the different methods at each surface pressure. It is clear that the trends for all surface pressures are similar. In addition, the χ^2 between each of the models and the experimental data for each contrast at an APM associated with a surface pressure of $30\,\mathrm{mN\,m^{-1}}$, the average χ^2 and standard deviation for each method are given in Table 4.3.

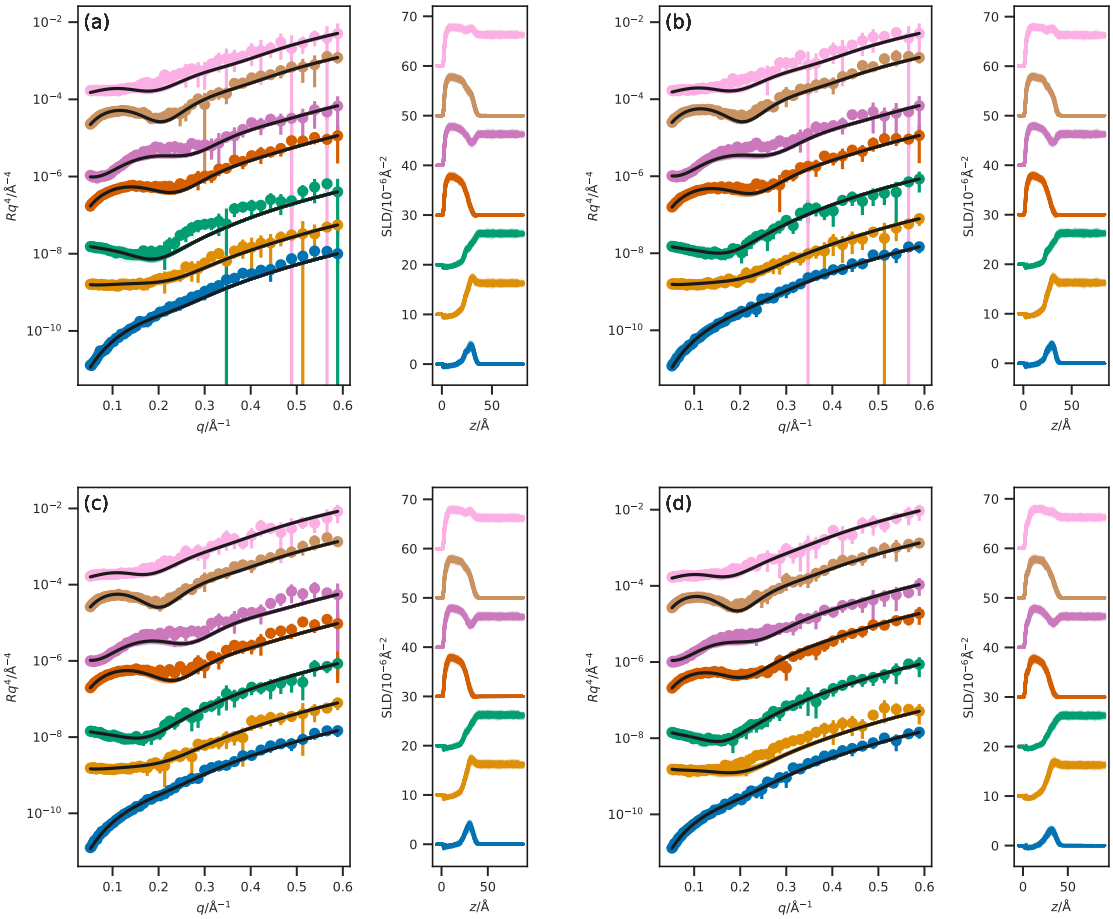


FIGURE 4.4: The NR profiles (left) and SLD profiles (right) determined from the Slipid all-atom potential model simulations of DSPC; (a) at $20\,\mathrm{mN\,m^{-1}}$, (b) at $30\,\mathrm{mN\,m^{-1}}$, (c) at $40\,\mathrm{mN\,m^{-1}}$, and (d) at $50\,\mathrm{mN\,m^{-1}}$. From top-to-bottom the contrasts are as follows; $d_8$3-D_2O, $d_8$3-ACMW, $d_7$0-D_2O, $d_7$0-ACMW, h-D_2O, $d_1$3-D_2O, $d_1$3-ACMW. The different contrast NR profiles have been offset in the y-axis by an order of magnitude and the SLD profiles offset in the y-axis by 10×10^{-6} Å^{-2}, for clarity.

TABLE 4.3: The χ^2 values for each of the reflectometry model at an APM associated with a surface pressure of $30\,\mathrm{mN\,m^{-1}}$.

Contrast	Chemically-consistent	Slipids	Berger	MARTINI
h-D_2O	27.93	166.19	119.72	1570.80
d_{13}-D_2O	84.83	235.50	90.35	2202.44
d_{13}-ACMW	86.25	92.21	85.29	200.48
d_{70}-D_2O	128.85	659.32	583.49	1965.42
d_{70}-ACMW	128.73	83.02	92.85	468.26
d_{83}-D_2O	298.33	343.03	424.55	3287.75
d_{83}-ACMW	97.29	144.51	315.74	910.49
Average	121.74 ± 78.65	246.25 ± 249.21	244.57 ± 238.79	1515.09 ± 1071.08

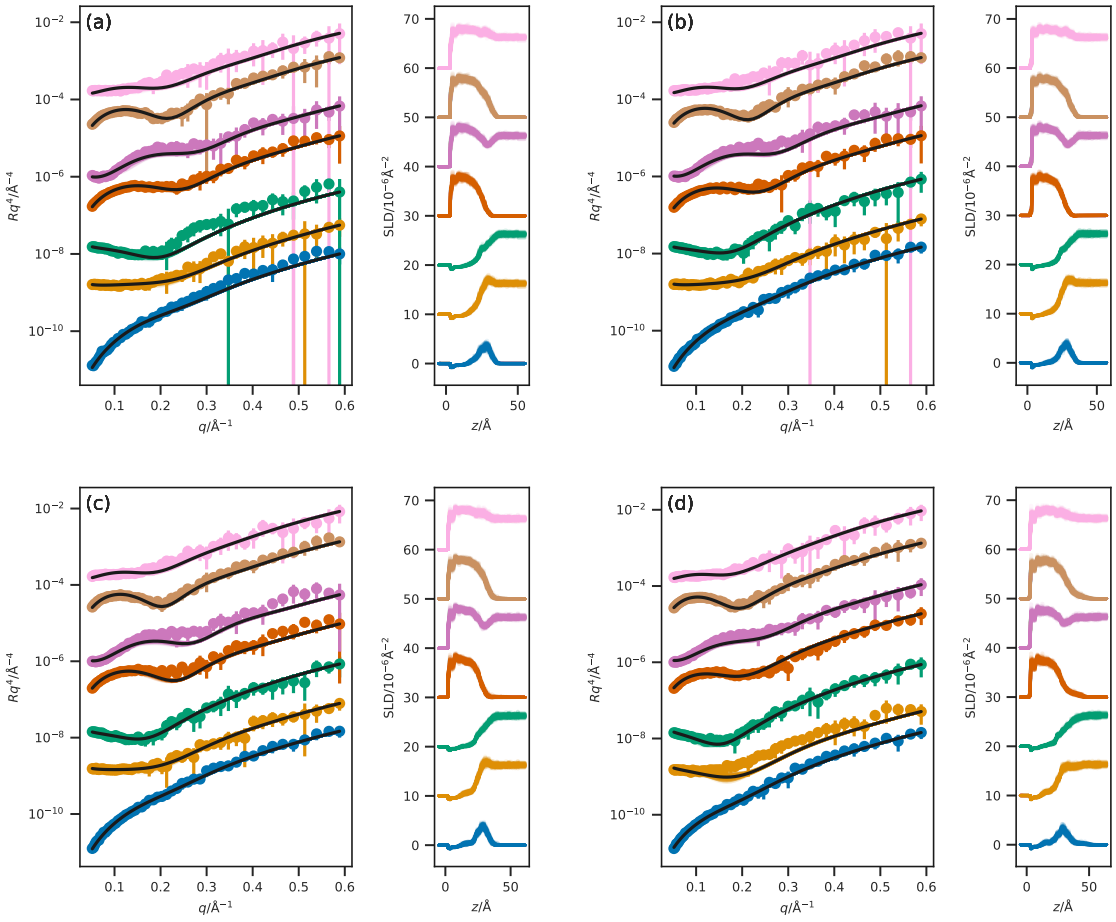


FIGURE 4.5: The NR profiles (left) and SLD profiles (right) determined from the Berger united-atom potential model simulations of DSPC; (a) at $20\,\mathrm{mN\,m^{-1}}$, (b) at $30\,\mathrm{mN\,m^{-1}}$, (c) at $40\,\mathrm{mN\,m^{-1}}$, and (d) at $50\,\mathrm{mN\,m^{-1}}$. From top-to-bottom the contrasts are as follows; $d_8$3-D_2O, $d_8$3-ACMW, $d_7$0-D_2O, $d_7$0-ACMW, h-D_2O, $d_1$3-D_2O, $d_1$3-ACMW. The different contrast NR profiles have been offset in the y-axis by an order of magnitude and the SLD profiles offset in the y-axis by 10×10^{-6} Å^{-2}, for clarity.

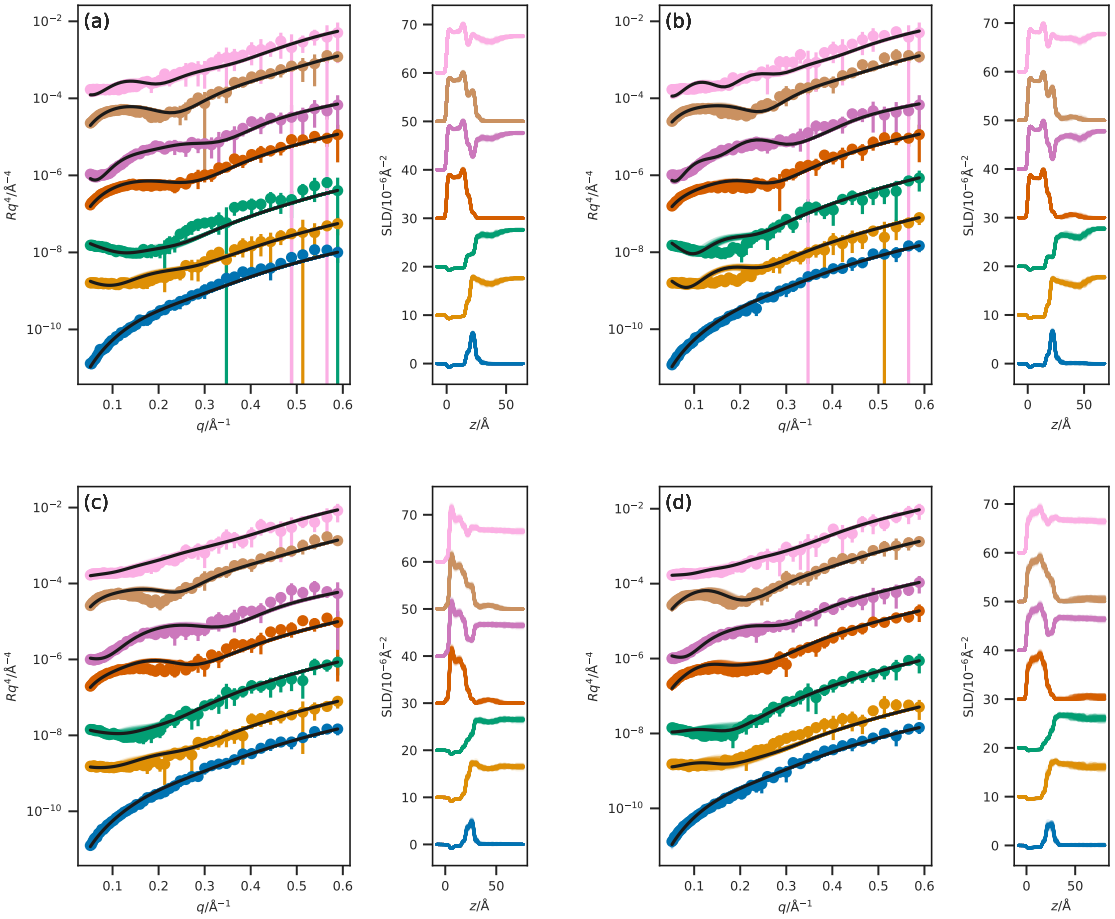


FIGURE 4.6: The NR profiles (left) and SLD profiles (right) determined from the MARTINI coarse-grained potential model simulations of DSPC; (a) at $20\,\text{mN}\,\text{m}^{-1}$, (b) at $30\,\text{mN}\,\text{m}^{-1}$, (c) at $40\,\text{mN}\,\text{m}^{-1}$, and (d) at $50\,\text{mN}\,\text{m}^{-1}$. From top-to-bottom the contrasts are as follows; $d_8$3-D_2O, $d_8$3-ACMW, $d_7$0-D_2O, $d_7$0-ACMW, h-D_2O, $d_1$3-D_2O, $d_1$3-ACMW. The different contrast NR profiles have been offset in the y-axis by an order of magnitude and the SLD profiles offset in the y-axis by $10 \times 10^{-6}\,\text{Å}^{-2}$, for clarity.

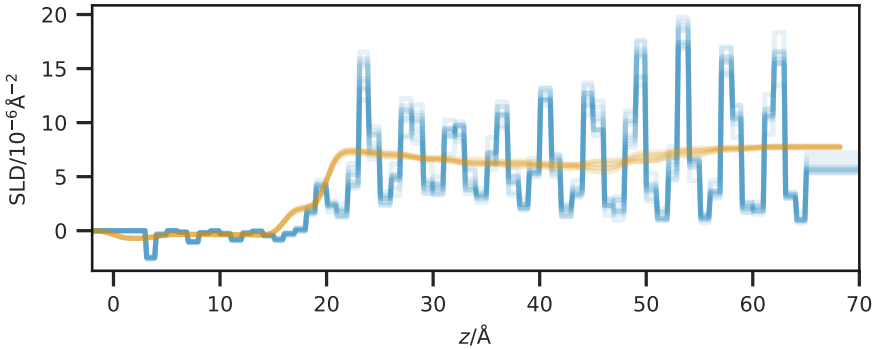


FIGURE 4.7: A comparison of the scattering length density profile for the MARTINI potential model simulations at an APM associated with a surface pressure of $30\,\mathrm{mN\,m^{-1}}$; the blue line shows the data where the layer thickness was 1 Å and no interfacial roughness and the orange line shows that with a layer thickness of 4 Å and a roughness of 0.2 Å

4.4.1 MARTINI

Initially, the MARTINI coarse-grained simulations were analysed with a layer thickness of 1 Å and an interfacial roughness of 0 Å, in a similar fashion to the other potential models. However, as can be seen in Figure 4.7 there is a clear ordering effect present in the MARTINI water, despite the use of the polarised water model. The effect of this ordering on the SLD profile, and therefore the reflectometry profile, can be reduced by using a larger layer thickness and introducing an interfacial roughness. Therefore, in the results discussed below the MARTINI potential model simulation were analysed using a layer thickness of 4 Å and an interfacial roughness of 0.2 Å. It is noted that this structuring may be reduced through the use of a less ordered wall [7] at the extreme of the simulation cell, however, the aim was to reproduce the experimental conditions using off-the-shelf tools and this would require custom modifications not easily available.

It can be seen from Table 4.3 that even with the larger layer thickness and adding interlayer roughness, the MARTINI potential model simulations do not effectively reproduce the reflectometry profile. Furthermore, it is noted that the agreement with the contrasts containing D_2O is particularly poor, this is most likely an artefact of the structuring effect mentioned above which cannot be completely removed.

However, it is noted that the agreement for the samples where the contrast uses ACMW, where the water is effectively removed from the SLD profile is also poor. This indicates that there are other artefacts limiting the applicability of the MARTINI potential model. Another such artefact is clear from investigating the calculated length of the hydrocarbon tail from the MARTINI simulation, at an APM associated with a surface pressure of $30\,\mathrm{mN\,m^{-1}}$, which was found to be $16.60^{+1.65}_{-1.88}$ Å, significantly less than the 24.3 Å estimated by the Tanford equation [36]. This is due to the nature of the MARTINI's 4-to-1 beading process, as DSPC has a hydrocarbon tail consisting of 18 carbon atoms, and it is not possible to bead such a chain accurately with the MARTINI potential model. In this work, a MARTINI phospholipid molecule was used with 4 MARTINI beads making up the chain; corresponding to an all-atom hydrocarbon chain of 16 atoms. Applying the Tanford equation to a hydrocarbon chain of such a length results in an anticipated length of 18.7 Å, which agrees better with that found from the simulation.

In addition to the disagreement from the tail beading process, there is also a clear problem with respect to the solvation the head layer by polarisable water beads. It can be seen that the number of water molecules per head group in the MARTINI potential model is typically 1.34 ± 0.35 , this is the value at an APM associated with a surface pressure of $30\,\mathrm{mN\,m^{-1}}$. The chemically-consistent model, however, gives a value of $5.49^{+0.86}_{-0.84}$. It is clear that the 4-to-1 beading present in the MARTINI potential model is creating water molecules that are too large to intercalate into the head layer structure, causing a reduction in the number of waters per head group.

The requirement for a 4-to-1 beading strategy for the MARTINI potential model is a significant weakness. A better method may be limiting experiments to a system that can be modelled exactly or the use of a different beading model. However, we are not aware of an off-the-shelf coarse-grained potential model that would easily offer the exact beading of DSPC.

4.4.2 Comparison with other simulations

Table 4.3 shows that both the Slipid and Berger potential model simulations agree well with the experimental data, with the Slipid potential model offering a slight improvement over the Berger. The quality of agreement between these higher-resolution potential models and the chemically-consistent model is relatively similar. However, the chemically-consistent model still offers a better fit to the experimental data than those determined from DM simulation.

The result that the chemically-consistent model offers better agreement with the data than those from even all-atom simulations is to be expected though, simply by considering the level of constraint present implicity when determining the reflectometry profile directly from the simulation. While the chemically-consistent constrains the layer model to ensure that the number of phospholipid head groups is the same as the pairs of tail groups, those from MD simulation have more realistic chemical constraints present from the potential model; e.g. bonding of atoms, and the non-bonded potentials. The quality of the agreement from this multi-modal approach is sufficient for such a method to be applied regularly to the analysis of neutron reflectometry.

Both the Slipid and Berger potential model simulations produced values for the tail length that were in better agreement with that from the Tanford equation than the MARTINI potential model simulations. For the Slipid potential model, with simulations at an APM associated with a surface pressure of $30\,\mathrm{mN\,m^{-1}}$ the tail length was found to be $20.17^{+1.41}_{-7.39}$ Å, while for the Berger potential model, at the same APM, a value of $19.80^{+1.59}_{-8.17}$ Å was obtained. Neither is quite as large as the 24.3 Å from the Tanford equation, however, it should be noted that this value is considered a theoretical maximum for a fully extended carbon chain.

Using the molecular dynamics simulations and the chemically-consistent model, it is possible to compare the number of water molecules per head group. From the Slipid and Berger potential model simulations, the number of water molecules per head group at an APM associated with a surface pressure of $30\,\mathrm{mN\,m^{-1}}$ was found to be $6.41^{+1.63}_{-0.76}$ and $5.49^{+0.68}_{-0.53}$ respectively. These are in good agreement with the value of $5.49^{+0.86}_{-0.84}$ found from the chemically-consistent model, using Equation 4.4.

It should be noted that to obtain the $50\,\mathrm{ns}$ production run simulation using the all-atom Slipid potential model required over 13 days of using 32 cores of the SCARF computing

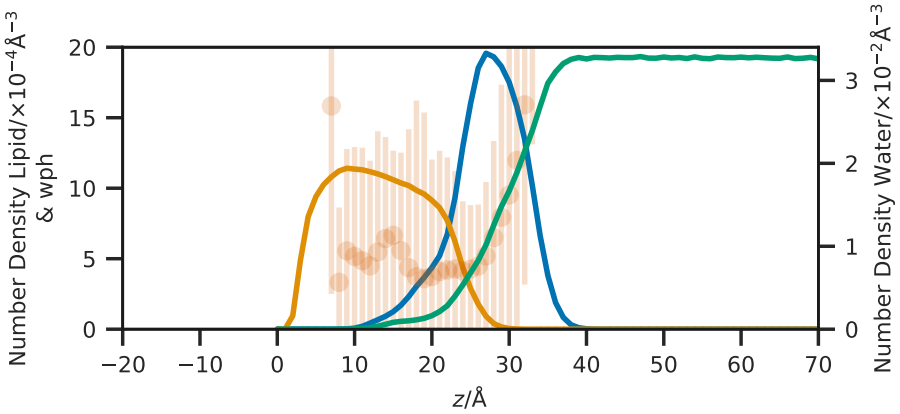


FIGURE 4.8: The number denisty for each component in the Slipids potential model simulation at an APM associated with a surface pressure of $30\,\mathrm{mN\,m^{-1}}$; heads (blue), tails (orange), water (green), and the number of water molucules per head group (yellow dots).

resource. This is non-trivial and therefore not necessarily applicable to all neutron reflectometry measurements. However, for the nearly as accurate Berger potential model simulations (which are only marginally less accurate), only approximately 2 days of the same compute resource was required. This suggests that given the quality of agreement with the experimental data achieved from, and compute requirement for, the Berger potential model simulations, such simulations could be run alongside experiments at large facilities to aid interpretation and analysis.

4.4.3 Using the Slipid potential model simulations to improve the monolayer model

Despite the chemically-consistent model offering a small improvement in agreement over the Slipid potential model simulation, we believe that it is possible to use the MD simulations to improve the existing this model. A possible improvement can be found from considering Figure 4.8, which shows the water molecules per head group as a function of depth; it is clear that solvent penetration within the phospholipid head layer is not uniform. In particular, there is a clear pocket of water formed around the ester group of the phospholipid. However, currently, the solvation is described as a uniform averaging of the head layer SLD profile with the SLD of the solvent. The structuring in the solvation suggests that perhaps a different model for solvent penetration could be applied to the head layer to more accurately describe the realistic solvent penetration.

It is generally suggested that the conformal roughness between the layers should be carried evenly through the monolayer when there is only a single phospholipid type [37]. However, from Figure 4.8, it is clear that the overlap between the head layer and water is much larger than that between the tail and head. Furthermore, there is additionally some solvent present deep within the tail layer, due to the presence of head group character within that region. While the interfacial roughness can be considered as describing the capillary wave from the water surface, it is also frequently applied to account for the intrinsic/structure roughness in the monolayer that exists due to the fact that the phospholipid ensemble comprises of molecules with different conformations and thus different chain lengths. This suggests that

perhaps the constraint that the interfacial roughness should be constant through the monolayer might be relaxed.

4.5 Conclusions

This chapter presents a direct comparison between a traditional method for the analysis of neutron reflectometry measurements with an analysis method using simulations from a range of all-atom and coarse-grained molecular dynamics potential models. It was shown that the MARTINI potential model could not accurately reproduce the experimental neutron reflectometry data, likely, due to the limitations of the 4-to-1 beading system when applied to a carbon chain of 18 atoms.

The Berger and Slipids potential models both showed good agreement with the experimental data, however, the best agreement was obtained from the traditional chemically-consistent layer model. This would be expected given that the chemically-consistent model contains many more "degrees of freedom" than the simulations which are severely chemically-constrained by the potential model.

Finally, some points from the highest resolution simulations were noted that may be used to improve the chemically-consistent modelling approach. For example, it is desirable to model the solvation of the phospholipid head groups in a non-uniform fashion. Application of these improvements may enable the more accurate modelling of phospholipid monolayers from neutron reflectometry.

4.6 References

[1] R. D. Barker, L. E. McKinley and S. Titmuss, in *Biophysics of Infection*, ed. M. C. Leake, Springer, New York, 2016, pp. 261–282.
[2] S. Khodakarimi, M. H. Hekmatshoar and F. Abbasi, *J. Mater. Sci. Mater. Electron.*, 2016, **27**, 182–190.
[3] S. Bobone, Y. Gerelli, M. De Zotti, G. Bocchinfuso, A. Farrotti, B. Orioni, F. Sebastiani, E. Latter, J. Penfold, R. Senesi, F. Formaggio, A. Palleschi, C. Toniolo, G. Fragneto and L. Stella, *(BBA) - Biomembranes*, 2013, **1828**, 1013–1024.
[4] A. F. Miller, M. R. Wilson, M. J. Cook and R. W. Richards, *Mol. Phys.*, 2003, **101**, 1131–1138.
[5] P. M. Anderson and M. R. Wilson, *J. Chem. Phys.*, 2004, **121**, 8503.
[6] L. Darré, J. Iglesias-Fernandez, A. Kohlmeyer, H. Wacklin and C. Domene, *J. Chem. Theory Comput.*, 2015, **11**, 4875–4884.
[7] A. Koutsioubas, *J. Phys. Chem. B*, 2016, **120**, 11474–11483.
[8] A. V. Hughes, F. Ciesielski, A. C. Kalli, L. A. Clifton, T. R. Charlton, M. S. P. Sansom and J. R. P. Webster, *Acta Crystallogr. D*, 2016, **72**, 1227–1240.
[9] A. P. Dabkowska, L. E. Collins, D. J. Barlow, R. Barker, S. E. McLain, M. J. Lawrence and C. D. Lorenz, *Langmuir*, 2014, **30**, 8803–8811.
[10] K. Pluhackova and R. A. Böckmann, *J. Phys. Condens. Matter*, 2015, **27**, 323103.
[11] S. J. Marrink, H. J. Risselada, S. Yefimov, D. P. Tieleman and A. H. de Vries, *J. Phys. Chem. B*, 2007, **111**, 7812–7824.
[12] O. Berger, O. Edholm and F. Jähnig, *Biophys. J.*, 1997, **72**, 2002–2013.
[13] J. J. Uusitalo, H. I. Ingólfsson, P. Akhshi, D. P. Tieleman and S. Marrink, *J. Chem. Theory Comput.*, 2015, **11**, 3932–3945.

[14] S. A. Sanders and A. Z. Panagiotopoulos, *J. Chem. Phys.*, 2010, **132**, 114902.
[15] S. Wang and R. G. Larson, *Langmuir*, 2015, **31**, 1262–1271.
[16] D. P. Tieleman, J. L. MacCallum, W. L. Ash, C. Kandt, Z. Xu and L. Monticelli, *J. Phys. Condens. Matter*, 2006, **18**, S1221–S1234.
[17] A. Cordomí, G. Caltabiano and L. Pardo, *J. Chem. Theory Comput.*, 2012, **8**, 948–958.
[18] J. P. M. Jämbeck and A. P. Lyubartsev, *J. Phys. Chem. B*, 2012, **116**, 3164–3179.
[19] J. P. M. Jämbeck and A. P. Lyubartsev, *J. Chem. Theory Comput.*, 2012, **8**, 2938–2948.
[20] J. P. M. Jämbeck and A. P. Lyubartsev, *J. Chem. Theory Comput.*, 2013, **9**, 774–784.
[21] E. Segala, D. Guo, R. K. Y. Cheng, A. Bortolato, F. Deflorian, A. S. Doré, J. C. Errey, L. H. Heitman, A. P. IJzerman, F. H. Marshall and R. M. Cooke, *J. Med. Chem.*, 2016, **59**, 6470–6479.
[22] Y. von Hansen, S. Gekle and R. R. Netz, *Phys. Rev. Lett.*, 2013, **111**, 118103–118105.
[23] C. M. Hollinshead, R. D. Harvey, D. J. Barlow, J. R. P. Webster, A. V. Hughes, A. Weston and M. J. Lawrence, *Langmuir*, 2009, **25**, 4070–4077.
[24] H. J. C. Berendsen, J. R. Grigera and T. P. Straatsma, *J. Phys. Chem.*, 1987, **91**, 6269–6271.
[25] S. O. Yesylevskyy, L. V. Schäfer, D. Sengupta and S. J. Marrink, *PLoS Comput. Biol.*, 2010, **6**, e1000810.
[26] H. J. C. Berendsen, D. van der Spoel and R. van Drunen, *Comput. Phys. Commun.*, 1995, **91**, 43–56.
[27] E. Lindahl, B. Hess and D. van der Spoel, *J. Mol. Model.*, 2001, **7**, 306–317.
[28] D. van der Spoel, E. Lindahl, B. Hess, G. Groenhof, A. E. Mark and H. J. C. Berendsen, *J. Comput. Chem.*, 2005, **26**, 1701–1718.
[29] B. Hess, C. Kutzner, D. van der Spoel and E. Lindahl, *J. Chem. Theory Comput.*, 2008, **4**, 435–447.
[30] I.-C. Yeh and M. L. Berkowitz, *J. Chem. Phys.*, 1999, **111**, 3155–3162.
[31] L. Martínez, R. Andrade, E. G. Birgin and J. M. Martínez, *J. Comput. Chem.*, 2009, **30**, 2157–2164.
[32] W. Humphrey, A. Dalke and K. Schulten, *J. Mol. Graph.*, 1996, **14**, 33–38.
[33] I. Kubo, S. Adachi, H. Maeda and A. Seki, *Thin Solid Films*, 2001, **393**, 80–85.
[34] A. R. J. Nelson and S. W. Prescott, *J. Appl. Crystallogr.*, 2019, **52**, 193–200.
[35] A. R. J. Nelson, S. W. Prescott, I. Gresham and A. R. McCluskey, *Refnx v0.1.2*, `http://doi.org/10.5281/zenodo.2552023`, 2019.
[36] C. Tanford, *The Hydrophobic Effect: Formation of Micelles and Biological Membranes*, John Wiley & Sons, New York, 2nd edn., 1980.
[37] R. A. Campbell, Y. Saaka, Y. Shao, Y. Gerelli, R. Cubitt, E. Nazaruk, D. Matyszewska and M. J. Lawrence, *J. Colloid Interface Sci.*, 2018, **531**, 98–108.

5 Assessing particle swarm methods to small angle scattering analysis

Abstract

Context

5.1 Introduction

Small angle scattering is a popular technique for the structural investigation of surfactant micelles [1]. Typically, coarse shape-based modelling (such as that introduced in Section 2.2.7) is used for the analysis. These shapes allow for the classification of the micelle shape and interactions. However, as with reflectometry, there has been a growing interest in the use of atomistic simulations as a multi-modal analysis tool for solution scattering methods [2].

The use of atomistic simulation as an analysis method in the study of biomolecules has been popular for many years [3–5]. These have built on the success of biomolecular simulation and crystallography, using structural information from the protein data bank [6] and applying popular all-atom potential models. Typically, this is used for the study of systems where the solution state differs significantly from that present in the crystal. Popular systems that this analysis is applied to are flexible protein multimers, the benefit of molecular dynamics being that an ensemble of structures can be represented in a single simulation trajectory [7, 8].

The importance of molecular dynamics simulation to represent an ensemble of structures has lead to the application of interesting aspect from probability theory. In particular, the use of Bayesian inference has been popular in understanding that presence of different structures in solution. For example, in the work of Bowerman *et al.* [8], accelerated molecular dynamics simulations (similar to traditional molecular dynamics however a 'boost' potential is applied to the system to improve sampling) were performed on an all-atom representation of the protein multimer tri-ubiquitin. The scattering profile was calculated from the simulation and evaluated against experimental data, and the agreement between the simulation and experiment assess in a Bayesian fashion, with a uniform prior probability. This methodology showed that the presence of a two state ensemble was more probable that the single state that would be obtained from, for example, a crystallographic study.

The simulation of a surfactant micelle is inherently more complex than for a protein ensemble, due to the greater number of states available under thermodynamic conditions. This means that there is rarely a suitable starting structure, instead it is necessary to simulation the system from a random configuration or artifically create a micelle structure as a starting position. Early work on the simulation of surfactant aggregation involved the $4.5\,\mathrm{ns}$ simulation of 42 sodium dodecyl sulfate (SDS) molecules [9]. These simulations started with a random solution of surfactant molecules and resulted in two small surfactant aggregates consisting of 17 and 25 molecules each. The aggregates simulated were much smaller than those measured experimentally by small angle neutron scattering, which were found to consist of $(79 \pm 1)\,\mathrm{m}$olecules [10]. However, this deviation may be due to the fact that the simulations were performed with simulation thermostat at $60\,^{\circ}\mathrm{C}$, and the SDS micelle size is noted to reduce with increasing temperature [11]. Maillet *et al.* studied the mechanism of micelle formation with a $3\,\mathrm{ns}$ simulation of the cationic *n*-nonyltrimethylammonium chloride; observing micelle formation, fragmentation, and monomer exchange [12]. The micelles that were formed were smaller than would be expected (between 15 and 20 molecules), however, this work suggested that the initial stages of micellisation process are dominated by collisions between aggregates, whereas monomer exchanges are more frequent closer to equilibrium.

These early examples of the simulation of a micellisation event both used all-atom potential models [9, 12], however, it is clear that in order to obtain an experimentally realistic micelle from simulation a longer simulation would be required. In Section 2.3.3, the use of

potential model coarse-graining as a method to increase the "real-time" length of a molecular dynamics simulation was introduced. Therefore, coarse-graining was used as a tool to simulate micellisation events from a random solution of surfactants. For example, in the work of Jorge [13], a united-atom model (where all hydrogen atoms are integrated into the atoms to which they are bound) for *n*-decyltrimethylammonium bromide was used to simulate micellisation at $80\,^{\circ}\mathrm{C}$. This work showed that the use of a united-atom model allow for a more efficient simulation, but also resulted in a larged mass average cluster size. However, again a value of only ~ 25 was reached after $14\,\mathrm{ns}$, as with the work of Tarek *et al.* [9], these small micelles might be realistic at these elevated temperatures, however, to my best knowledge no experimental structural data, such as SANS measurements, exists for *n*-decyltrimethylammonium bromide micelles at $80\,^{\circ}\mathrm{C}$. Sanders and Panagiotopoulos attempted to use the MARTINI coarse-grained potential model to simulate the micellisation of the zwitterionic dodecylphosphocholine (DPC). This was simulated at $97\,^{\circ}\mathrm{C}$ for $1.8\,\mu\mathrm{s}$ resulting in a trajectory where the cluster size mode was 41, the authors noted that experimentally at $25\,^{\circ}\mathrm{C}$ a micelle size of 56 ± 5 is expected. Again, it was noted that this variation may be due to the increased simulation temperature [14, 15].

The use of mesoscale simulation techniques, such as dissipative particle dynamics (DPD), are common for the simulation of surfactant molecules in solution [16]. DPD simulations are similar to coarse-grained molecular dynamics simulation, however, with the inclusion additional dissipative and random forces. These serve both as a thermostat and to make up for degrees of freedom lost to coarse-graining, meaning that sites within a DPD model typically account for more atoms than in even a MARTINI coarse-grained molecular dynamics simulation. An example of the use of DPD to study micellisation is that of Vishnyakov *et al.* [17], where the critical micelle concentration (cmc) of nonionic surfactants was investigated. This work found cmc values and mean micelle aggregation numbers that agreed quantitatively with experimental measurements. While mesoscopic techniques, such as DPD, are an interesting tool for the study of rough micelle strucutre, these methods lack the structural details that make their use exciting as a method of the multimodal analysis of small angle scattering data.

It is clear from the discussion above that the simulation of experimentally realistic micelles from a random structure would require significantly longer simulations than are currently available. Therefore a more pragmatic approach is necessary, essentially this can take one of two routes; building a micelle-like starting structure based on a simple analysis of the experimental data or tackling the problem as an optimisation challange, where the aim is to optimise the atomistic (or near-atomistic) structure to the experimental data. The former method was that applied by Ivanovic *et al.* in their recent work [2], where simulations were performed and compared with experimental scattering of micelles of *n*-dodecyl-β-*D*-maltoside (DDM) and *n*-decyl-β-*D*-maltoside (DM). This work used two approaches to determine the size of the micelle that should be simulated; the first used the forward scattering (that at $I(0)$) to determine the density and therefore the aggregation number of the micelle [18], while the second involved the simulation of a series of micelles of different sizes and the calculation of scattered intensity at a single scattering vector. These methods gave good agreement as to the size of the micelle, and the scattering profiles calculated from $50\,\mathrm{ns}$ molecular dynamics simulations offered reasonable agreement with those measured experimentally at a series of temperatures. The experimental scattering profiles where then used as an energetic restraint on the simulation improving the agreement between experiment and simulation.

The work of Ivanovic *et al.* benefited from the monodispersity of the particular micellar system choosen. However, if a more polydisperse system were being studied, the *pragmatic*

approach may require significantly more computation, as many more aggregation numbers of surfactants would need to be considered. In this chapter, I discuss a truely-model free approach for the analysis of micellar small angle scattering data, by applying an global-optimisation process a near-atomistic system. This involved using a particle swarm algorithm to fit to an experimental scattering profile, while a energy minimisation of the potential model was performed using a simple gradient descent approach. It was believed that the application of a population-based optimisation method would result in a series of suitable structures, allowing for a more realistic understanding of the ensemble structures that are present for micelles in solution.

5.2 Methods

The computational methodology described herein has been implemented in the open-source C_MPI program `fitoog` [**?**].

5.2.1 Simulation Methodology

The `fitoog` software takes as input a series of input files that define the molecules and the intramolecular interactions. The molecule input file is space separated consisting of an index, particle name, x-coordinate, y-coordinate, z-coordinate, and scattering length. While, the intramolecular interactions file describes the bonds within the molecule, in lieu of angular interactions a pseudo-bond is defined between each second particle in the system. When the initial molecule input file is read in, a differences object is created which stores the differential between the particle description and the more severe coarse-grained description that is used in the particle swarm optimisation.

In order to reduce the parameter space that was probed by the particle swarm algorithm, the description of the surfactants as severely coarse-grained. This involved reducing the surfactant to a 'director' description, that is where the surfactant molecule is reduced to a position and direction, shown pictorially in Figure 5.1. The use of this description allowed the surfactant to be desrcibed by just six variables; three of which described the position of the centre-of-mass position of the molecule (a, b, c) and three that describe the orientaion of the surfactant in space (ϕ, ω, κ). These variables were those that were exposed to the particle swarm optimisation method outlined in Section 2.4.2.

Following each iteration of the particle swarm algorithm, the director description of the surfactant was expanded (using the differences object mentioned above) to a MARTINI representation (or whatever representation was used as an input). The molecule was then rotated based on a rotation matrix, in this work the rotation matrix was constructed by first rotating the rotation axis by $-\phi$ and $-\omega$, then rotating by κ in around the z-axis, before rotating the axis back to the original position by ω and ϕ [**?**] (Figure **??** defines these angles). With the severely coarse-grained description fully expanded, the system could be subjected to a potential model energy minimisation algorithm (implemented as a gradient descent) if desired before the scattering profile was calculated using the Golden Vectors method developed by Watson and Curtis [19]. The agreement between the calculated scattering profile and the 'experimental' scattering was used as a figure of merit, ζ, that was to be optimised by the particle swarm algorithm.

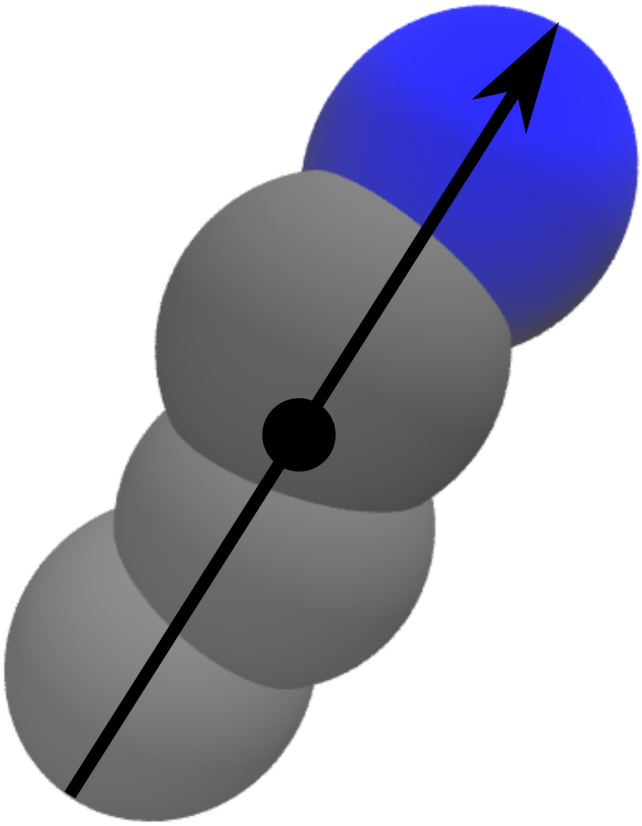


FIGURE 5.1: A graphical description of the severe coarse-graining applied to the MARTINI description of the *n*-decyltrimethylammonium surfactant molecule for the use of the particle swarm algorithm.

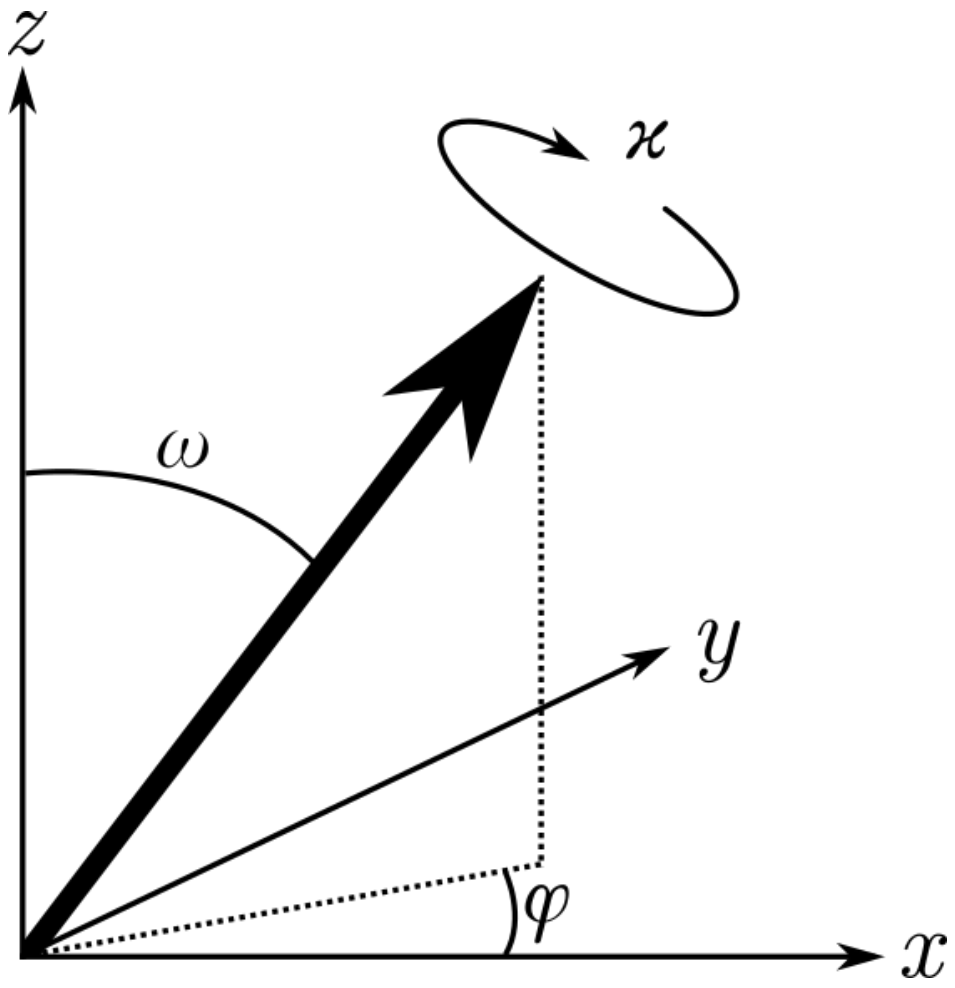


FIGURE 5.2: The definitation of the polar angles used in the coarse grained representation of the surfactant molecule.

FIGURE 5.3: A schematic of the parallelisation methodology applied currently in the `fitoog` code.

5.2.2 Parallelisation

The use of a population based optimisation methods allowed for easy access to highly parallel simulation. This was achieved but parallelisation over the cores available to the simulation such that a population is split evenly across them. There is intercore messaging that makes use of the MPI libraries, where the figure of merit from each core is shared with a parent core that determines the global best population. This global best is then shared to all cores of the process. Figure 5.3 described the parallelisation method pictorially. The parallelisation methodologies is to reduce the time spent when the helper cores are not in use. I feel that, while more sophisticated methodologies exist (such having a core-level best population and only occasionally finding the true global best), this implementation offers a useful test-case for assessing the utility of this optimisation methodology.

5.3 Results & Discussion

5.4 Conclusions

5.5 References

[1] A. Sanchez-Fernandez, T. Arnold, A. J. Jackson, S. L. Fussell, R. K. Heenan, R. A. Campbell and K. J. Edler, *Phys. Chem. Chem. Phys.*, 2016, **18**, 33240–33249.
[2] M. T. Ivanović, L. K. Bruetzel, J. Lipfert and J. S. Hub, *Angew. Chemie Int. Ed.*, 2018, **57**, 5635–5639.
[3] S. J. Perkins, A. S. Nealis, B. J. Sutton and A. Feinstein, *J. Mol. Biol.*, 1991, **221**, 1345–1366.
[4] M. O. Mayans, W. J. Coadwell, D. Beale, D. B. A. Symons and S. J. Perkins, *Biochem. J.*, 1995, **331**, 283–291.
[5] J. S. Hub, *Curr. Opin. Struct. Biol.*, 2018, **49**, 18–26.
[6] *RCSB PDB: Protein Data Bank*, `http://www.rcsb.org`, (accessed 2018-01-28).
[7] P.-C. Chen and J. S. Hub, *Biophys. J.*, 2014, **107**, 435–447.
[8] S. Bowerman, A. S. J. B. Rana, A. Rice, G. H. Pham, E. R. Strieter and J. Wereszczynski, *J. Chem. Theory Comput.*, 2017, **13**, 2418–2429.
[9] M. Tarek, S. Bandyopadhyay and M. L. Klein, *J. Mol. Liq.*, 1998, **78**, 1–6.
[10] P. A. Hassan, G. Fritz and E. W. Kaler, *J. Colloid Interface Sci.*, 2003, **257**, 154–162.
[11] S. Hayashi and S. Ikeda, *J. Phys. Chem.*, 1980, **84**, 744–751.
[12] J.-B. Maillet, V. Lachet and P. V. Coveney, *Phys. Chem. Chem. Phys.*, 1999, **1**, 5277–5290.
[13] M. Jorge, *Langmuir*, 2008, **24**, 5714–5725.
[14] A. Malliaris, J. Le Moigne, J. Sturm and R. Zana, *J. Phys. Chem.*, 1985, **89**, 2709–2713.
[15] N. Kamenka, Y. Chevalier and R. Zana, *Langmuir*, 1995, **11**, 3351–3355.
[16] J. C. Shelley and M. Y. Shelley, *Curr. Opin. Colloid Interface Sci.*, 2000, **5**, 101–110.
[17] A. Vishnyakov, M.-T. Lee and A. V. Neimark, *J. Phys. Chem. Lett.*, 2013, **4**, 797–802.
[18] J. Lipfert, L. Columbus, V. B. Chu, S. A. Lesley and S. Doniach, *J. Phys. Chem. B*, 2007, **111**, 12427–12438.
[19] M. C. Watson and J. E. Curtis, *J. Appl. Crystallogr.*, 2013, **46**, 1171–1177.

6 Developing open-source teaching resources for classical simulation and scattering

Abstract

Classical molecular dynamics simulations are a common component of multi-modal analyses from scattering measurements, such as small-angle scattering and reflectometry. Users of these experimental techniques often have no formal training in the theory and practice of molecular dynamics simulation, leading to the possibility of these simulations being treated as a "black box" analysis technique. In this chapter, two open educational resources (OERs); `pylj` and "*The Interaction Between Simulation and Scattering*" are described and their utility to introducing users of scattering methods to the tools of molecular dynamics discussed. These resources are a Python library designed to allow users to interact with molecular dynamics simulations and a series of interactive web pages, to introduce classical simulation and how it may be applied to scattering.

Context

Often users of scattering methods are interested in the use of classical simulation to better understand the results that they have obtained. However, regularly those with an experimental background aim to apply molecular simulation methods without true consideration of the difficulty of this. These OERs were developed to introduce experimental colleague (and in the case of `pylj` undergraduate students) to classical simulation and molecular dynamics. They reduce the cognitive load necessary to understand these methods by providing a simple, but mathematically rigorous interface and enabling the "worked example effect" throughout.

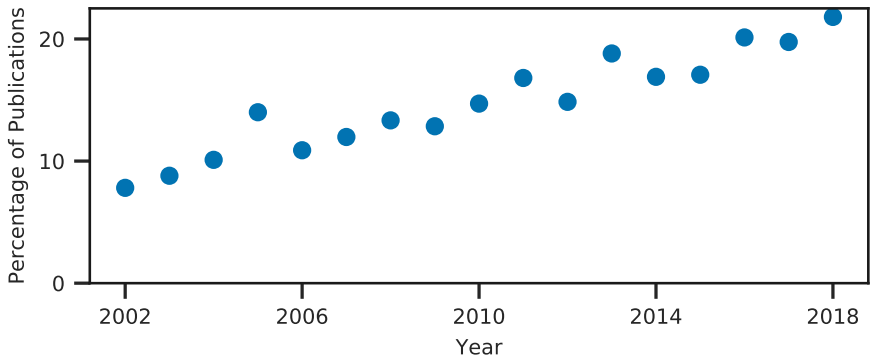


FIGURE 6.1: The annual percentage of publications that mention "small angle scattering" that also mention "molecular dynamics", determined from the numbers of matching Goggle Scholar results.

6.1 Introduction

The popularity of classical simulation, both all-atom and coarse-grained, as a technique for multi-modal analysis of scattering techniques, such as reflectometry and small angle scattering, has grown linearly over the past two decades [1–6]. Figure 6.1 shows that as of 2019, $\sim$20 % of all small angle scattering publications also mention molecular dynamics. Users of scattering techniques often have backgrounds in experimental science and may have received little formal training in the theory or practice of computational modelling. This can lead to the use of molecular dynamics simulations as a "black box" without necessarily understanding the underlying methodologies, or considering possible sources of error. To help support researchers use molecular dynamics simulations in their analysis of scattering data while reducing the risk of modelling errors, a number of software tools, such as WAXSiS and SASSIE [7–9] have been developed that present easy-to-use, graphical, web-based user interfaces.

A complementary approach is to organise educational activities, such as lectures and workshops, tailored to introduce molecular simulation techniques to audiences of scattering users. One example is the annual ISIS Neutron Training Course, which includes a module titled "An Introduction to Molecular Dynamics for Neutron Scattering". This module covers the fundamentals of classical molecular dynamics simulation, presents applications of these methods in neutron science, and gives students practical hands-on experience with the SASSIE software package [9].

While lectures and workshops are an effective tool for education and training, participation can be limited due to difficulties attending in person (due to location and cost) or physical limits on student numbers. An alternative educational strategy gaining popularity within scientific and engineering communities is the publication of open educational resources (OERs). These are courses, lectures, or learning resources published online that are freely available for use by anyone. In addition to their broad accessibility, these resources have permissive "open" copyright licenses that allow their use in the "5R activities": retain, reuse, revise, remix, and redistribute [10]. Publishing a resource as an OER increases the reach and impact, because others may use it in their own teaching not only in its original form, but are free to modify, and redistribute, the material to better suit their aims. The OERs developed

as a part of this work both leveraged heavily the Jupyter Notebook framework [11] to enable technology-enhanced OERs.

6.1.1 Using Jupyter Notebooks in Education

Project Jupyter [11] is a collection of standards, a community, and a set of software tools. The Jupyter Notebook is one of these software tools that is capable of creating, editing, and running a Jupyter Notebook file. This is a file that can contain executable code (in this work this is exclusively in the Python programming language) and narrative text (either Markdown or formatted LaTex), enabling the user to "tell an interactive, computational story" [12]. Furthermore, the interactive nature reduces the barrier of entry to computational methods that is often imposed on those learning, due often to the need to understand a command line interface.

The Jupyter Notebook framework has become a popular platform for OERs that teach computational skills, because it allows authors to include instructional text, images and other media, alongside the executable, editable code, in an example of "literate programming" [13]. This format encourages students to directly interact with code examples by running, editing, and rerunning these within the source document [14], supporting exploratory experiential learning [15], and enabling the "worked example effect" [16]. Furthermore, the modular nature of a Jupyter Notebook OER allows the resource designer to build computational tools to be used by those learning, such as Python libraries to aid understanding.

6.1.2 Teaching computational simulation

It was suggested by Aiello-Micosia and Sperandeo-Mineo [17] that the understanding of the microscopic disordered motions of particles in gases is a difficult problem for many science students. While Pallant and Tinker [18] comment that there was an educational challenge associated with helping students to rationalise the relationship between the mathematics underlying computational simulation and the behaviour of the system. However, it has been noted that the visualisation often used in the traditional teaching of molecular dynamics simulations may cause difficulties for students' understanding, that can be categorised as follows [19]:

- visual subtlety: often simulations are presented as two-dimensional displays of three-dimensional objects, creating spatial relationships that may be difficult to interpret,
- complexity: high information depth in an image, perhaps of a complex chemical model, will lead to increases, often unnecessarily in cognitive load,
- abstractness and conceptual depth: conventions are often used to represent phenomena that may be vague or unfamiliar to those learning about the methods.

Therefore, in addition to making use of software that will enable learners to interact with the computational methods being introduced, it is important that the discussion and visualisation of these methods are as straightforward as possible.

CODE BLOCK 6.1: An example of an NVT ensemble molecular dynamics algorithm as implemented in `pylj`.

```
from pylj import md, sample


def md_simulation(
    number_of_particles, temperature, box_length, number_of_steps
):
    """
    A simple NVT ensemble molecular dynamics simulation implemented in the
    pylj package

    Parameters
    ----------
    number_of_particles: int
        Number of particles to be simulated
    temperature: float
        Temperature of thermostat (Kelvin)
    box_length: float
        Size of the periodic cell (Angstrom)
    number_of_steps: int
        Number of molecular dynamics iterations

    Returns
    -------
    System
        A pylj class containing all inform from the simulation
    """
    system = md.initialise(
        number_of_particles, temperature, box_length, "square"
    )
    sample_system = sample.Energy(system)
    system.time = 0
    for i in range(0, number_of_steps):
        system.integrate(md.velocity_verlet)
        system.md_sample()
        system.heat_bath(temperature)
        system.time += system.timestep_length
        system.step += 1
        if system.step % 10 == 0:
            sample_system.update(system)
    return system
```

6.2 `pylj`: an open-source teaching tool for classical atomistic simulation

`pylj` (PYthon Lennard-Jones) [20, 21] is an educational software and molecular dynamics engine designed to introduce students to the details of classical simulation. Initially, `pylj` was only able to utilise a Lennard-Jones potential model [22], however, recently (as of version 1.1.0 [23]) there is the ability to include any custom forcefield, with a Buckingham potential packaged with the software [24]. In an effort to reduce the complexity and visual subtlety of typical molecular dynamics simulations, pylj performs two-dimension molecular dynamics or Monte-Carlo simulation. From the beginning, this software was designed to operate in the Jupyter Notebook framework and therefore eliminate the need for the learner to interact with the command line interface, as is the case with common molecular dynamics packages like Gromacs [25], LAMMPS [26], or DL_POLY [27].

6.2.1 Software design

`pylj` was designed such that it may be operated at a series of different levels of abstraction. For example, an educator could write a simple function to allow the running of a molecular dynamics simulation (Code Block 6.1) or an interested student could manually interact with the source code. This abstraction is achieved through the modular design of pylj, where the `md.py` or `mc.py` modules implement all of the functionality related to a particular simulation method. The simulation controlled by an overarching `System` class, which contains all of the information regarding the simulation that has been/is running.

CODE BLOCK 6.2: The Lennard Jones potential model as implemented in pylj.

```
import numpy as np


def lennard_jones(dr, constants, force=False):
    """Calculate the energy or force for a pair of particles using the
    Lennard-Jones (A/B variant) forcefield.

    Parameters
    ----------
    dr: float, array_like
        The distances between the all pairs of particles.
    constants: float, array_like
        An array of length two consisting of the A and B parameters for
        the 12-6 Lennard-Jones function.
    force: bool (optional)
        If true, the negative first derivative will be found.
    Returns
    -------
    float:
        The potential energy or force between the particles.
    """
    if force:
        return 12 * constants[0] * np.power(dr, -13) - (
            6 * constants[1] * np.power(dr, -7)
        )
    else:
        return constants[0] * np.power(dr, -12) - (
            constants[1] * np.power(dr, -6)
        )
```

The computationally intensive nature of the pairwise force and energy calculations necessary for molecular dynamics and Monte Carlo simulation necessitated the use of Cython, a method of including compiled C language code within a Python package [28]. This enabled a $\sim$10 times speed up in the determination of the pairwise interactions when compared with the use of pure Python implementation. From version 1.2.1 [29], the pure Python versions of the pairwise interactions built using `numba` just-in-time compilation [30], which enabled a 5 times speed up. This means that a `pylj` simulation without the compile C functions is now just twice as slow than the compiled version. Furthermore, this speed is now comparable to the length of time taken to render the visualisation, meaning that the pairwise interactions are no longer the rate determining step in the use of `pylj`.

The `sample.py` module is integral to the utility of `pylj`, as this allows educators to easily create custom visualisation environments. Using this module, is it possible to create a molecular dynamics simulation that enables for the plotting of a huge variety of outputs. Plots such as instantaneous pressure against time, radial distribution function, and instantaneous temperature histogram are included in `pylj` and it is easy for the users to create custom plots or build visualisations where multiple plots can be presented together.

Currently, the `forcefield.py` module is the location where the potential model may be defined. This involves the definition of a single function that describes the potential model (Code Block 6.2 gives the function for the Lennard-Jones potential model), with a boolean flag that defines where the force or energy should be returned. In future, for a planned `pylj` 2.0, the potential model definition will be adapted such that each model is an individual class containing functions for each of the different parameters to be calculated. This will more easily allow the growth of `pylj` to enable features such as the simulation or particles with different potential model parameterisations and the inclusion of mixing rules.

6.2.2 Applications

In order to give an idea of the capabilities of the `pylj` software, three typical applications are discussed, a further application is evident in Section 6.3.

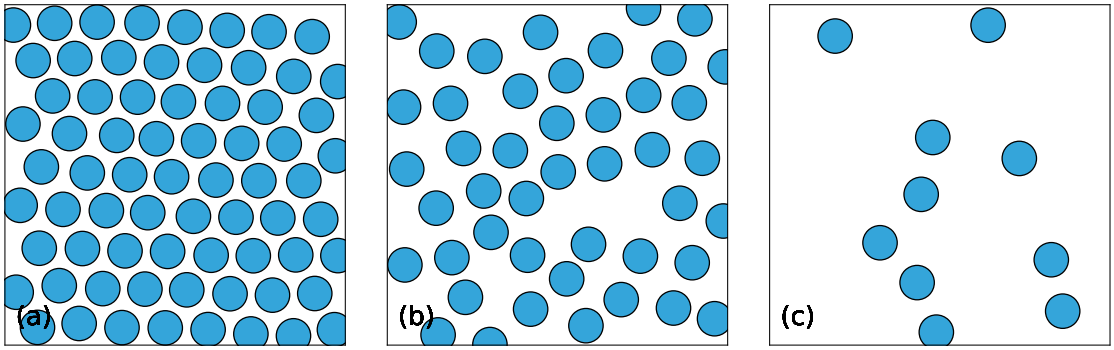


FIGURE 6.2: A snapshot of a `pylj` molecular dynamics simulation for: (a) a solid, (b) a liquid, and (c) a gas.

States of matter

States of matter (solid, liquid, and gas) is a common high school level science subject. Where the different states of matter are rationalised in terms of the atomic density and interactions. Often this is introduced with pictorial examples showing a two-dimensional representation of a hexagonally close-packed crystal, a disordered liquid, and a low-density gas with the atoms represented as circular particles. `pylj` is capable of easily reproducing these diagram (Figure 6.2), while increasing student engagement by representing a "real" chemical system in thermal motion. `pylj` was recently used by Dr Benjamin Morgan of the University of Bath for a demonstration such as this in a seminar introducing chemical simulation to a cohort of mathematicians.

Ideal gas law

When `pylj` was originally published [20], the repository included an example of a possible laboratory exercise where the ideal gas law was modelled using molecular dynamics simulation. This was achieved by varying the particle density and measuring the time-averaged pressure of the simulation. At low particle densities, where the interactions of the particles are unlikely, the `pylj` molecular dynamics simulation agrees well with the ideal gas law. However, as the particle density increases such that the interparticle interactions are more frequent, deviations are observed in agreement with the van der Waals equation (Figure 6.3). Using this exercise, it is possible to introduce a cohort of students to the insight available from chemical simulation, without a significant focus on the simulation methods increasing the accessibility to students in the first or second year of their undergraduate.

Molecular dynamics & Monte Carlo

The final application of `pylj` is the use for it in teaching molecular dynamics or Monte Carlo methods. The framework of `pylj` means that it is straightforward, and does not necessarily require substantial familiarity with the Python programming language. Code Block 6.1 shows a molecular dynamics algorithm, while Code Block 6.3 gives that for canonical Monte Carlo. It is clear that both of this algorithm as simple and clear to implement, allowing the focus of the laboratory exercise to be on the students' understanding of the methodology. `pylj` has been applied in this way in a third-year undergraduate laboratory exercise at the University of Bath which introduced students to molecular dynamics simulations.

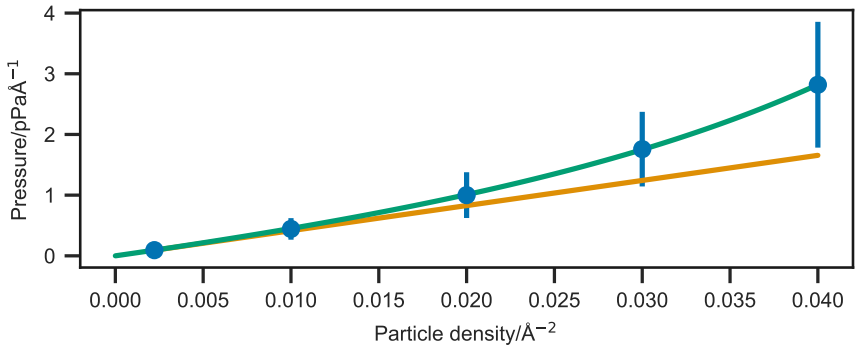


FIGURE 6.3: The deviation from the ideal gas law observed using 0.1 ns `pylj` simulations (blue circle), the ideal gas law is shown with a solid orange line, which the van der Waals equation of state is shown with a solid green line.

CODE BLOCK 6.3: An example of an canonical Monte Carlo algorithm as implemented in `pylj`.

```
def mc_simulation(
    number_of_particles, temperature, box_length, number_of_steps
):
    """
    A simple NVT ensemble Monte Carlo simulation implemented in the
    pylj package

    Parameters
    ----------
    number_of_particles: int
        Number of particles to be simulated
    temperature: float
        Temperature of thermostat (Kelvin)
    box_length: float
        Size of the periodic cell (Angstrom)
    number_of_steps: int
        Number of Monte Carlo iterations

    Returns
    -------
    System
        A pylj class containing all inform from the simulation
    """
    system = mc.initialise(
        number_of_particles, temperature, box_length, "square"
    )
    sample_system = sample.Energy(system)
    system.compute_energy()
    system.old_energy = system.energies.sum()
    system.mc_sample()
    for i in range(0, number_of_steps):
        system.step += 1
        system.select_random_particle()
        system.new_random_position()
        system.compute_energy()
        system.new_energy = system.energies.sum()
        if mc.metropolis(
            temperature, system.old_energy, system.new_energy
        ):
            system.accept()
        else:
            system.reject()
        system.mc_sample()
        if system.step % 10 == 0:
            sample_system.update(system)
    return system
```

6.3 The interaction between simulation and scattering

The Jupyter Notebook framework and `pylj` software we then used to enable learning and understanding in the OER entitled "The interaction between simulation and scattering" (available at pythoninchemistry.org/sim_and_scat) [31]. This is an online, open-source, interactive learning resource written to introduce members of the scattering and diffraction community to molecular dynamics simulations. The aim is to improve their understanding, and therefore reduce the treatment of molecular dynamics as "black box" calculation by experimental colleagues. The OER comprises five lessons that introduce classical molecular dynamics methods and show how these can be used to assist in the analysis of experimental scattering data by the calculation of a simulation scattering profile from the molecular dynamics simulation. `pylj` is used to provide simple, but computationally authentic, examples of simulations, that demonstrate visually and programmatically the conceptual relationships between simulation and scattering techniques. Finally, we show a "real-world" example of calculation of the scattering profile from a simulation of a lysozyme protein in solution.

6.3.1 Resource construction

The resource is available in two main formats. First, as a series of web pages, hosted at pythoninchemistry.org/sim_and_scat. Secondly, as the source-code repository used to build these webpages [31]. The source content consists of a set of Jupyter Notebooks and Markdown files, which are automatically compiled using the `jupyter-book` tool [32] to generate the web version. This system allows the resulting webpages to include text, equations, and figures, which describe key concepts and explain details of algorithms, as well as Python code blocks, which provide specific examples. The web pages have Thebelab and BinderHub integrations [33–35], which allow students to launch interactive versions of these webpages that allow execution and modification of the included Python code. The ability to read the resource as an "interactive document" improves the ability for the students to engage in the "worked example effect" [16]. The resource is provided under a CC-BY license [36], while the `jupyter-book` software is shared under an MIT license [37], both of which are open and highly permissinve. This allows readers to reuse or remix the material to enhance their own educational platform and for secondary authors to contribute to improving the source material.

6.3.2 Resource outline

The resource follows a simple outline that introduces key aspects of molecular dynamics simulations.

Home

The welcome page introduces the resource, explains the purpose, and gives the user information about how the resource may be used, including details of the Thebelab and BinderHub integrations. This page also contained details of the permitted use/reuse, sharing of the content of the resource and licensing. Finally, a list of authors and contributors if given.

Classical methods

This section introduces concepts related to classical simulation methods. Including the use of interatomic potential functions, alongside some examples, such as the Lennard-Jones and Buckingham potential models [22, 24]. The problem of parameterising a potential model is then suggested, showing that the use of higher accuracy quantum mechanical calculations to do so. The presence of off-the-shelf, general potential models are discussed; with the caveat that they may still require system-specific optimisation. Finally, we mention mixing rules; again discussing the possible problems that a user may encounter if applying these blindly to specific systems.

Molecular dynamics

With the concept of a classical interatomic potential introduced, the resource then begins to focus on how these are used in molecular dynamics simulations. We work through how a one dimensional NVE (constant number, volume, and energy) molecular dynamics simulation may be built, using the Velocity-Verlet algorithm and the Lennard-Jones potential model [22, 38]. The Velocity-Verlet algorithm is introduced in terms of Newton's laws of motion and the generalised equations of motion. Finally, we discuss a range of key factors that can affect molecular dynamics simulations; simulation ensembles, the distance cut-off for an interatomic potential, and the use of periodic boundary conditions.

`pylj` and interaction with scattering

The final aspect of the resource covers using molecular dynamics simulations to understand scattering profiles. This is presented as a practical example, using the `pylj` package [20, 21]. A two-dimensional molecular dynamics simulation of argon atoms interacting through a Lennard-Jones potential is demonstrated. The users are first shown this working `pylj` simulation and invited to interact with the simulation and the custom plotting functionality of `pylj`. The concept of a radial distribution function (RDF) is then shown, and the user is given the opportunity to run some `pylj` simulations with the RDF being output alongside the simulation window. Next, the Debye equation [39] is present and it is shown how it may be used to calculate scattering data from a simulation. The user is invited to observe the effect of simulation temperature on the resulting scattering profile. We finish by discussing alternative, faster, algorithms for calculating scattering profile, such as the Fibonacci Sequence or Golden Vectors method [40, 41]

"Real" simulation and scattering

Having shown the development of a scattering profile from an idealised system, we then direct the user to a popular resource for the GROMACS [25] molecular dynamics software. This resource gives a quick introduction to using GROMACS to simulate a lysozyme molecule in buffer [42], the student may then use their own simulated trajectory or one that can be downloaded from the OER. We show how the system may be visualised, introduce the `MDAnalysis` Python package for the analysis of molecular dynamics trajectories [43, 44], and show the scattering profile developed from the lysozyme simulation compared with experimental data [45]. The module finished by pointing the student to resources to more easily resolve scattering data from the molecular simulation, such as SASSIE

and CRYSOL [9, 46]. The focus of this resource is to introduce simulation methodologies to users of scattering to aid their understanding, not to derive the exact mechanics of the calculation fo scattering from a simulation. Resources for this purpose already exist and have well-developed tutorials, so it is not necessary to recreate such software here.

6.4 Conclusions

In the chapter, two open educational resources focussed on classical simulation and molecular dynamics were shown. The first of these resources was the `pylj` Python package, which designed for use at any education level to give an easy, visual example of classical simulation. This software is open-source and actively developed, with the growth of capability and applications in the future. Currently, the code is used in the third year computational chemistry laboratory at the University of Bath and there is an ongoing discussion for it to be used in future in the second year computational chemistry laboratory at the University of Bristol. Additionally, the webpage for pylj at pythoninchemistry.org/pylj has been viewed over 400 times since launching in June 2018.

The second OER is the online, interactive learning module for the introduction of users of experimental scattering methods to classical simulation. This module is shared under an open, permissive license and in future hope that its use/reuse will be uptaken by educators of scattering worldwide. Furthermore, there is scope to introduce the module as a flipped learning component [47] within scattering courses at the ISIS Neutron and Muon Source and Diamond Light Source.

6.5 References

[1] J. Pan, F. A. Heberle, S. Tristram-Nagle, M. Szymanski, M. Koepfinger, J. Katsaras and N. Kučerka, *BBA - Biomembranes*, 2012, **1818**, 2135–2148.
[2] L. Boldon, F. Laliberte and L. Liu, *Nano Rev.*, 2015, **6**, 25661.
[3] J. S. Hub, *Curr. Opin. Struct. Biol.*, 2018, **49**, 18–26.
[4] A. Koutsioubas, *J. Phys. Chem. B*, 2016, **120**, 11474–11483.
[5] L. Darré, J. Iglesias-Fernandez, A. Kohlmeyer, H. Wacklin and C. Domene, *J. Chem. Theory Comput.*, 2015, **11**, 4875–4884.
[6] E. Scoppola and E. Schneck, *Curr. Opin. Colloid Interface Sci.*, 2018, **37**, 88–100.
[7] P.-C. Chen and J. S. Hub, *Biophys. J.*, 2014, **107**, 435–447.
[8] C. J. Knight and J. S. Hub, *Nucleic Acids Res.*, 2015, **43**, W225–W230.
[9] S. J. Perkins, D. W. Wright, H. Zhang, E. H. Brookes, J. Chen, T. C. Irving, S. Krueger, D. J. Barlow, K. J. Edler, D. J. Scott, N. J. Terrill, S. M. King, P. D. Butler and J. E. Curtis, *J. Appl. Crystallogr.*, 2016, **49**, 1861–1875.
[10] D. Wiley, *Open Content - Definition*, `http://opencontent.org/definition`, (accessed 2018-11-19).
[11] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla and C. Willing, *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, 2016, 87–90.
[12] L. A. Barba, L. J. Barker, D. Blank, J. Brown, A. Downey, T. George, L. Heagy, K. Mandli, J. Moore, D. Lippert, K. E. Niemeyer, R. Watkins, R. H. West, E. Wickes, C. Willing and M. Zingale, *Teaching and Learning with Jupyter*, `https://jupyter4edu.github.io/jupyter-edu-book/`, 2019, (accessed 2019-03-04).
[13] D. E. Knuth, *Comput. J.*, 1984, **27**, 97–111.

[14] L. A. Barba, A. Wickenheiser and R. Watkins, *CyberTraining: DSE–The Code Maker: Computational Thinking for Engineers with Interactive Contextual Learning*, https://doi.org/10.6084/m9.figshare.5662051.v1, 2017.
[15] S. Papert, *Mindstroms*, Basic Books, New York City, US, 1993.
[16] R. A. Tarmizi and J. Sweller, *J. Educ. Psycho.*, 1988, **80**, 424–436.
[17] M. L. Aiello-Nicosia and R. M. Sperandeo-Mineo, *Eur. J. Phys.*, 1985, **6**, 148–153.
[18] A. Pallant and R. F. Tinker, *J. Sci. Educ. Technol.*, 2004, **13**, 51–66.
[19] L. L. Jones, K. D. Jordan and N. A. Stillings, *Chem. Educ. Res. Pract.*, 2005, **6**, 136–149.
[20] A. R. McCluskey, B. J. Morgan, K. J. Edler and S. C. Parker, *J. Open Source Educ.*, 2018, **1**, 19.
[21] A. R. McCluskey, A. R. Symington, B. J. Morgan, K. J. Edler and S. C. Parker, *Arm61/Pylj: Pylj-1.2.5*, http://doi.org/10.5281/zenodo.2587898, 2019.
[22] J. E. Lennard-Jones, *Proc. Royal Soc. Lond. A.*, 1924, **106**, 463–477.
[23] A. R. McCluskey, B. J. Morgan, K. J. Edler and S. C. Parker, *Arm61/Pylj: Pylj-1.1.0*, http://doi.org/10.5281/zenodo.1403828, 2018.
[24] R. A. Buckingham, *Proc. Royal Soc. Lond. A.*, 1938, **168**, 264–283.
[25] H. J. C. Berendsen, D. van der Spoel and R. van Drunen, *Comput. Phys. Commun.*, 1995, **91**, 43–56.
[26] S. Plimpton, *J. Comput. Phys.*, 1995, **117**, 1–19.
[27] W. Smith, C. W. Yong and P. M. Rodger, *Mol. Simulat.*, 2002, **28**, 385–471.
[28] *Cython 3.0a0*, https://cython.readthedocs.io/en/latest/, (accessed 2019-03-04).
[29] A. R. McCluskey, A. R. Symington, B. J. Morgan, K. J. Edler and S. C. Parker, *Arm61/Pylj: Pylj-1.2.1*, http://doi.org/10.5281/zenodo.2423866, 2019.
[30] *Numba: A High Performance Python Compiler*, http://numba.pydata.org, (accessed 2019-03-04).
[31] A. R. McCluskey, J. Grant, A. R. Symington, T. Snow, J. Doutch, B. J. Morgan, S. C. Parker and K. J. Edler, *Pythoninchemistry/Sim_and_scat: Sim_and_scat-v0.3-Preprint*, http://doi.org/10.5281/zenodo.2556824, 2019.
[32] S. Lau and C. Holdgraf, *Jupyter/Jupyter-Book Beta-2*, https://github.com/jupyter/jupyter-book, 2019.
[33] B. Ragan-Kelley, N. M. Thiéry, S. Corlay, S. Gutsche, C. Holdgraf and T. Head, *Minrk/Thebelab v0.3.3*, https://github.com/minrk/thebelab, 2019.
[34] B. Ragan-Kelley, Y. Panda, C. Willing, C. Holdgraf, K. Erdogan, T. Head, M. Bussonnier, H. Zarea, M. van Niekerk, C. Yick and E. Sundell, *Jupyterhub/Binderhub FirstLight*, https://github.com/jupyterhub/binderhub, 2019.
[35] Jupyter and others, Proceedings of the 17th Python in Science Conference., 2018.
[36] C. Commons, *Creative Commons – Attribution 4.0 International – CC-BY-4.0*, https://creativecommons.org/licenses/by/4.0/, 2019.
[37] O. Source, *The MIT License*, https://opensource.org/licenses/MIT, 2019.
[38] W. C. Swope, H. C. Andersen, P. H. Berens and K. R. Wilson, *J. Chem. Phys.*, 1982, **76**, 637–649.
[39] P. Debye, *Ann. Phys.*, 1915, **351**, 809–823.
[40] M. C. Watson and J. E. Curtis, *J. Appl. Crystallogr.*, 2013, **46**, 1171–1177.
[41] D. I. Svergun, *Acta Crystallogr. A*, 1994, **50**, 391–402.
[42] J. A. Lemkul, *GROMACS Tutorial: Lysozyme in Water*, http://www.mdtutorials.com/gmx/lysozyme/index.html, (accessed 2019-03-08).
[43] N. Michaud-Agrawal, E. J. Denning, T. B. Woolf and O. Beckstein, *J. Comput. Chem.*, 2011, **32**, 2319–2327.
[44] R. Gowers, M. Linke, J. Barnoud, T. Reddy, M. Melo, S. Seyler, J. Domański, D. Dotson, S. Buchoux, I. Kenney and O. Beckstein, Python in Science Conference, Austin, Texas,

2016, pp. 98–105.
[45] D. Franke, C. M. Jeffries and D. I. Svergun, *Nat. Methods*, 2015, **12**, 419–422.
[46] D. Svergun, C. Barberato and M. H. J. Koch, *J. Appl. Crystallogr.*, 1995, **28**, 768–773.
[47] *Flipped Learning | Higher Education Academy*, `https://www.heacademy.ac.uk/knowledge-hub/flipped-learning-0`, (accessed 2019-03-12).

7 Summary & Future Work